PTO/SB/123 (04-15)
Approved for use through 03/31/2021. OMB 0651-0035
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.

CHANGE OF CORRESPONDENCE ADDRESS
Patent

Address to:
Mail Stop Post Issue
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Patent Number	11681614
Issue Date	06/20/2023
Application Number	17531506
Filing Date	09/12/2022
First Named Inventor	Andrey V. Kuzmin
Attorney Docket Number	RMS-03K2 / 373Z

Please change the Correspondence Address for the above-identified patent to:

☐ The address associated with Customer Number: [ ]

OR

☒ **Firm *or* Individual Name** Radian Memory Systems, LLC

6275 West Plano Parkway, Suite 500

Address

City Plano | **State** Texas | **ZIP** 75093

Country

Telephone | **Email**

This form cannot be used to change the data associated with a Customer Number. To change the data associated with an existing Customer Number use "Request for Customer Number Data Change" (PTO/SB/124).

This form will not affect any "fee address" provided for the above-identified patent. To change a "fee address" use the "Fee Address Indication Form" (PTO/SB/47).

I am the:

☐ Patentee.

☐ Assignee of record of the entire interest. See 37 CFR 3.71.
Statement under 37 CFR 3.73(b) is enclosed. (Form PTO/SB/96).

☒ Attorney or agent of record. Registration Number 35675.

Signature /Marc P. Schuyler/

Typed or Printed Name Marc P. Schuyler

Date 12/03/2024 | Telephone

NOTE: Signatures of all the inventors or assignees of record of the entire interest or their representative(s) are required. Submit multiple forms if more than one signature is required, see below*.

☐ *Total of _____ forms are submitted.

This collection of information is required by 37 CFR 1.33. 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 3 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: Mail Stop Post Issue, 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.

Privacy Act Statement

The **Privacy Act of 1974 (P.L. 93-579)** requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (*i.e.*, GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC ACKNOWLEDGEMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/931,506	**12/03/2024 04:48:48 PM Z ET**	**2024024 / RMS03K1-323Z**

Title of Invention

STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	11681614
CONFIRMATION #	3894	FILED BY	Marc Schuyler
PATENT CENTER #	68207657	FILING DATE	09/12/2022
CUSTOMER #	73091	FIRST NAMED INVENTOR	Andrey V. Kuzmin
CORRESPONDENCE ADDRESS	-	AUTHORIZED BY	-

Documents

TOTAL DOCUMENTS: 1

DOCUMENT	PAGES	DESCRIPTION	SIZE (KB)
sb0123_fill-323Z-17931506-3894.pdf	2	Change of Address	68 KB

Digest

DOCUMENT	MESSAGE DIGEST(SHA-512)
sb0123_fill-323Z-17931506-3894.pdf	430BD6DDFC3DD016AB2FDB1F1BB55D40890B0434D8BE3E306DEAA10B9DE7D3B2133915532AA41C9B96F9227E50307DB9A6A00C731FDD9303FD7EDF8110C8C4A5

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 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.

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	FILING OR 371(C) DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
17/931,506	09/12/2022	Andrey V. Kuzmin	RMMS.003C12

CONFIRMATION NO. 3894

73091
Marc P. Schuyler
P.O. Box 2535
Saratoga, CA 95070

POA ACCEPTANCE LETTER

OC000000081382158

Date Mailed: 12/03/2024

NOTICE OF ACCEPTANCE OF POWER OF ATTORNEY

This is in response to the Power of Attorney filed 11/22/2024.

The Power of Attorney in this application is accepted. Correspondence in this application will be mailed to the above address as provided by 37 CFR 1.33.

Questions about the contents of this notice and the requirements it sets forth should be directed to the Office of Data Management, Application Assistance Unit, at **(571) 272-4000** or **(571) 272-4200** or **1-888-786-0101**.

/qnguyen/

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	FILING OR 371(C) DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
17/931,506	09/12/2022	Andrey V. Kuzmin	RMMS.003C12

CONFIRMATION NO. 3894

POWER OF ATTORNEY NOTICE

OC000000081382116

20995
Knobbe, Martens, Olson & Bear, LLP
2040 MAIN STREET
FOURTEENTH FLOOR
IRVINE, CA 92614

Date Mailed: 12/03/2024

NOTICE REGARDING CHANGE OF POWER OF ATTORNEY

This is in response to the Power of Attorney filed 11/22/2024.

- The Power of Attorney to you in this application has been revoked by the assignee who has intervened as provided by 37 CFR 3.71. Future correspondence will be mailed to the new address of record(37 CFR 1.33).

Questions about the contents of this notice and the requirements it sets forth should be directed to the Office of Data Management, Application Assistance Unit, at **(571) 272-4000** or **(571) 272-4200** or **1-888-786-0101**.

/qnguyen/

PTO/AIA/80 (07-12)
Approved for use through 11/30/2014. OMB 0651-0035
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.

POWER OF ATTORNEY TO PROSECUTE APPLICATIONS BEFORE THE USPTO

I hereby revoke all previous powers of attorney given in the application identified in the attached statement under 37 CFR 3.73(c).

I hereby appoint:

☒ Practitioners associated with Customer Number: 73091

OR

☐ Practitioner(s) named below (if more than ten patent practitioners are to be named, then a customer number must be used):

Name	Registration Number	Name	Registration Number

As attorney(s) or agent(s) to represent the undersigned before the United States Patent and Trademark Office (USPTO) in connection with any and all patent applications assigned only to the undersigned according to the USPTO assignment records or assignments documents attached to this form in accordance with 37 CFR 3.73(c).

Please change the correspondence address for the application identified in the attached statement under 37 CFR 3.73(c) to:

☒ The address associated with Customer Number: 73091

OR

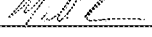
☐	Firm or Individual Name			
	Address			
	City		State	Zip
	Country			
	Telephone		Email	

Assignee Name and Address: Radian Memory Systems, Inc., 5010 N. Parkway Calabasas Unit 205, Calabasas, CA 91302

A copy of this form, together with a statement under 37 CFR 3.73(c) (Form PTO/AIA/96 or equivalent) is required to be Filed in each application in which this form is used. The statement under 37 CFR 3.73(c) may be completed by one of The practitioners appointed in this form, and must identify the application in which this Power of Attorney is to be filed.

SIGNATURE of Assignee of Record

The individual whose signature and title is supplied below is authorized to act on behalf of the assignee

Signature		Date	
Name	Mike Jadon	Telephone	
Title	Officer, Radian Memory Systems, Inc.		

This collection of information is required by 37 CFR 1.31, 1.32 and 1.33. 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 3 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/AIA/80 (07-17)
Approved for use through 03/31/2021. OMB 0651-0035
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no person is required to respond to a collection of information unless it displays a valid OMB control number

POWER OF ATTORNEY TO PROSECUTE APPLICATIONS BEFORE THE USPTO

I hereby revoke all previous powers of attorney given in the application identified in the attached statement under 37 CFR 3.73(c).

I hereby appoint:

[X] Practitioners associated with Customer Number: 73091

OR

[ ] Practitioner(s) named below (if more than ten patent practitioners are to be named, then a customer number must be used):

Name	Registration Number

Name	Registration Number

As attorney(s) or agent(s) to represent the undersigned before the United States Patent and Trademark Office (USPTO) in connection with any and all patent applications assigned only to the undersigned according to the USPTO assignment records or assignment documents attached to this form in accordance with 37 CFR 3.73(c).

Please change the correspondence address for the application identified in the attached statement under 37 CFR 3.73(c) to:

[X] The address associated with Customer Number: 73091

OR

[ ]

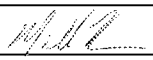
Firm or individual name		
Address		
City	State	Zip
Country		
Telephone	Email	

Assignee name and address: RADIAN MEMORY SYSTEMS LLC
C/O FORTRESS INVESTMENT GROUP
1345 AVENUE OF THE AMERICAS, 46TH FLOOR
NEW YORK, NEW YORK 10105

A copy of this form, together with a statement under 37 CFR 3.73(c) (Form PTO/AIA/96 or equivalent) is required to be filed in each application in which this form is used. The statement under 37 CFR 3.73(c) may be completed by one of the practitioners appointed in this form, and must identify the application in which this Power of Attorney is to be filed.

SIGNATURE of Assignee of Record

The individual whose signature and title is supplied below is authorized to act on behalf of the assignee.

Signature 	Date 13 June 2024
Name Mike Jadon	Telephone
Title Mike Jadon, CEO, Radian Memory Systems, LLC	

This collection of information is required by 37 CFR 1.31, 1.32, and 1.33. The information is required to obtain or retain a benefit by the public, which is to update (and by the USPTO to process) the file of a patent or reexamination proceeding. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and 1.14. This collection is estimated to take 18 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.

Privacy Act Statement

The **Privacy Act of 1974 (P.L. 93-579)** requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (*i.e.*, GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

PTO/AIA/96 (08-12)
Approved for use through 01/31/2013. OMB 0651-0031
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.

STATEMENT UNDER 37 CFR 3.73(c)

Applicant/Patent Owner: Radian Memory Systems, Inc. / Radian Memory Systems, LLC

Application No./Patent No.: 17931506 / USPT 11681614 Filed/Issue Date: 09/12/2022 Fd; 06/20/2023 ID

Titled: STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

Radian Memory Systems, LLC, a Corporation

(Name of Assignee) (Type of Assignee, e.g., corporation, partnership, university, government agency, etc.)

states that, for the patent application/patent identified above, it is (choose **one** of options 1, 2, 3 or 4 below):

1. ☑ The assignee of the entire right, title, and interest.

2. ☐ An assignee of less than the entire right, title, and interest (check applicable box):

   ☐ The extent (by percentage) of its ownership interest is _____%. Additional Statement(s) by the owners holding the balance of the interest must be submitted to account for 100% of the ownership interest.

   ☐ There are unspecified percentages of ownership. The other parties, including inventors, who together own the entire right, title and interest are:

   Additional Statement(s) by the owner(s) holding the balance of the interest must be submitted to account for the entire right, title, and interest.

3. ☐ The assignee of an undivided interest in the entirety (a complete assignment from one of the joint inventors was made). The other parties, including inventors, who together own the entire right, title, and interest are:

   Additional Statement(s) by the owner(s) holding the balance of the interest must be submitted to account for the entire right, title, and interest.

4. ☐ The recipient, via a court proceeding or the like (*e.g.*, bankruptcy, probate), of an undivided interest in the entirety (a complete transfer of ownership interest was made). The certified document(s) showing the transfer is attached.

The interest identified in option 1, 2 or 3 above (not option 4) is evidenced by either (choose **one** of options A or B below):

A. ☐ An assignment from the inventor(s) of the patent application/patent identified above. The assignment was recorded in the United States Patent and Trademark Office at Reel _____, Frame _____, or for which a copy thereof is attached.

B. ☑ A chain of title from the inventor(s), of the patent application/patent identified above, to the current assignee as follows:

1. From: Andrey V. Kuzmin/James G. Wayda To: Radian Memory Systems, LLC

   The document was recorded in the United States Patent and Trademark Office at
   Reel 031357, Frame 0503, or for which a copy thereof is attached.

2. From: Radian Memory Systems, LLC To: Radian Memory Systems, Inc.

   The document was recorded in the United States Patent and Trademark Office at
   Reel 040579, Frame 0405, or for which a copy thereof is attached.

This collection of information is required by 37 CFR 3.73(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/AIA/96 (08-12)
Approved for use through 01/31/2013. OMB 0651-0031
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.

STATEMENT UNDER 37 CFR 3.73(c)

3. From: Radian Memory Systems, Inc. To: Radian Memory Systems, LLC

The document was recorded in the United States Patent and Trademark Office at
Reel 067471, Frame 0240, or for which a copy thereof is attached.

4. From: _____ To: _____

The document was recorded in the United States Patent and Trademark Office at
Reel _____, Frame _____, or for which a copy thereof is attached.

5. From: _____ To: _____

The document was recorded in the United States Patent and Trademark Office at
Reel _____, Frame _____, or for which a copy thereof is attached.

6. From: _____ To: _____

The document was recorded in the United States Patent and Trademark Office at
Reel _____, Frame _____, or for which a copy thereof is attached.

☐ Additional documents in the chain of title are listed on a supplemental sheet(s).

☑ As required by 37 CFR 3.73(c)(1)(i), the documentary evidence of the chain of title from the original owner to the assignee was, or concurrently is being, submitted for recordation pursuant to 37 CFR 3.11.

[NOTE: A separate copy (i.e., a true copy of the original assignment document(s)) must be submitted to Assignment Division in accordance with 37 CFR Part 3, to record the assignment in the records of the USPTO. See MPEP 302.08]

The undersigned (whose title is supplied below) is authorized to act on behalf of the assignee.

/Marc P. Schuyler/	22 November 2024
Signature	Date
Marc P. Schuyler	35675
Printed or Typed Name	Title or Registration Number

Privacy Act Statement

The **Privacy Act of 1974 (P.L. 93-579)** requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (*i.e.*, GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC ACKNOWLEDGEMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/931,506	**11/22/2024 08:14:09 PM Z ET**	**RMMS.003C12**

Title of Invention

STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	11681614
CONFIRMATION #	3894	FILED BY	Marc Schuyler
PATENT CENTER #	68096604	FILING DATE	09/12/2022
CUSTOMER #	20995	FIRST NAMED INVENTOR	Andrey V. Kuzmin
CORRESPONDENCE ADDRESS	-	AUTHORIZED BY	-

Documents

TOTAL DOCUMENTS: 3

DOCUMENT	PAGES	DESCRIPTION	SIZE (KB)
5-aia0096-AssnStatement-2024-11-22-Kn01.pdf	3	Assignee showing of ownership per 37 CFR 3.73	120 KB
6-NEWPOA-RadianLLC-aia0080-sig.pdf	2	Power of Attorney	564 KB
6-aia0080-POA-2016-11-08-s.pdf	1	Power of Attorney	43 KB

Digest

DOCUMENT	MESSAGE DIGEST(SHA-512)
5-aia0096-AssnStatement-2024-11-22-Kn01.pdf	43FD4FAA454A10F726C17A9EF8A0739CEAE1C10B28CF4DEAFB634F684BD83B0BB844A922700B11D1E4262FCDEFCB1161

	C94374D2428D0E28F76DFD04542B3C92
6-NEWPOA-RadianLLC-aia0080-sig.pdf	E1B2C2B5485C9F090C4022A241762AA5DA909874B5E28ADC7CCDA791EDBDEB2D799D7A2DBC4E86E8AEA1AC7C3FB503525F46E92654ADC9F6E708F0D11ED5C11E
6-aia0080-POA-2016-11-08-s.pdf	D0DABD9FD1E247BD2A7A8696EFAFFBF004DBF91FF0AC5BD3F59B5DEFA05D3C56B7533DCDCAE21C8A8EB50B66B61374FD109CCD0336BCF8050FF3A3579F94C3BA

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 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.

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

ACKNOWLEDGEMENT OF LOSS OF ENTITLEMENT TO ENTITY STATUS DISCOUNT

APPLICATION #	FILING OR 371(C) DATE	FIRST NAMED APPLICANT	ATTORNEY DOCKET #	REQUEST ID
17/931,506	09/12/2022	Andrey V. Kuzmin	RMMS.003C12	219696

The entity status change request below filed through Patent Center on 05/23/2024 has been accepted.

Certifications

APPLICANT CHANGING TO REGULAR UNDISCOUNTED FEE STATUS

Signature

I certify, in accordance with 37 CFR 1.4(d)(4), that I am one of the signatories making the entity status change.

Signature	Name	Registration #
/Aaron Davis/	Aaron Davis	62027

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, VA 22313-1450
www.uspto.gov

APPLICATION NO.	ISSUE DATE	PATENT NO.
17/931,506	20-JUN-23	11681614

KNOBBE MARTENS OLSON & BEAR LLP
2040 MAIN STREET FOURTEENTH FLOOR
IRVINE, CA 92614

EGRANT NOTIFICATION

Your electronic patent grant (eGrant) is now available, which can be accessed via Patent Center at https://patentcenter.uspto.gov

The electronic patent grant is the official patent grant under 35 U.S.C. 153. For more information, please visit https://www.uspto.gov/electronicgrants

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.	ISSUE DATE	PATENT NO.	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/931,506	06/20/2023	11681614	RMMS.003C12	3894

20995 7590 05/31/2023

KNOBBE MARTENS OLSON & BEAR LLP
2040 MAIN STREET
FOURTEENTH FLOOR
IRVINE, CA 92614

ISSUE NOTIFICATION

The projected patent number and issue date are specified above. The patent will issue electronically. The electronically issued patent is the official patent grant pursuant to 35 U.S.C. § 153. The patent may be accessed on or after the issue date through Patent Center at https://patentcenter.uspto.gov/. The patent will be available in both the public and the private sides of Patent Center. Further assistance in electronically accessing the patent, or about Patent Center, is available by calling the Patent Electronic Business Center at 1-888-217-9197.

The USPTO is implementing electronic patent issuance with a transition period, during which period the USPTO will mail a ceremonial paper copy of the electronic patent grant to the correspondence address of record. Additional copies of the patent (i.e., certified and presentation copies) may be ordered for a fee from the USPTO's Certified Copy Center at https://certifiedcopycenter.uspto.gov/index.html. The Certified Copy Center may be reached at (800)972-6382.

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

The Patent Term Adjustment is 0 day(s). Any patent to issue from the above-identified application will include an indication of the adjustment on the front page.

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 Application Assistance Unit (AAU) of the Office of Patents Stakeholder Experience (OPSE), Stakeholder Support Division (SSD) at (571)-272-4200.

INVENTOR(s) (Please see PAIR WEB site http://pair.uspto.gov for additional inventors):

Andrey V. Kuzmin, Moscow, RUSSIAN FEDERATION;
James G. Wayda, Laguna Niguel, CA;

APPLICANT(s) (Please see PAIR WEB site http://pair.uspto.gov for additional applicants):

Radian Memory Systems, Inc., Manhattan Beach, CA;

The United States represents the largest, most dynamic marketplace in the world and is an unparalleled location for business investment, innovation, and commercialization of new technologies. The USA offers tremendous resources and advantages for those who invest and manufacture goods here. Through SelectUSA, our nation works to encourage and facilitate business investment. To learn more about why the USA is the best country in the world to develop technology, manufacture products, and grow your business, visit SelectUSA.gov.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventors: Andrey V. Kuzmin, et al.	**Conf. No.:** 3894
App. No.: 17/931,506	**Filed:** September 12, 2022
Examiner: Bataille, Pierre Miche	**Art Unit:** 2136
Title: STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS	

COMMENTS ON STATEMENT OF REASONS FOR ALLOWANCE

Mail Stop: Issue Fee
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Commissioner:

Applicant appreciates the finding of the pending claims to be allowable. In the Notice of Allowance dated January 10, 2023, the Examiner provides reasons for allowance of the claims.

Applicant assumes that, pursuant to M.P.E.P. § 1302.14, the Examiner has stated some, but not all of the reasons for allowance of the claims. As a result, the statements noted above do not necessarily relate to or completely set out all the reasons for allowance of each and every claim. Further, Applicant notes that each of the claims is patentable based on the combination of elements recited therein and not on any particular feature or features.

Applicant respectfully notes that the patentability of each claim is derived from the claim as a whole as written and the reasons for allowance are not limited to features explicitly discussed by the Examiner or Applicant or the lack of such features in the prior art. To the extent that the Examiner refers to claimed subject matter with language that does not match actual claim language, Applicant notes that the scope of each claim is defined by the actual claim language as written. The independent claims recite different language and, therefore, can be patentable for different reasons. Furthermore, the patentability of each dependent claim does not solely depend upon the patentability of the respective independent claim from which it depends. Rather, each dependent claim recites a further distinguishing feature and combination of particular utility.

The scope of the present claims is defined by the claims as written and is not limited by the Examiner's brief comments.

Respectfully submitted,

KNOBBE, MARTENS, OLSON & BEAR, LLP

Dated: May 4, 2023

By: /Aaron M. Davis/
Aaron M. Davis
Registration No. 62,027
Registered Practitioner
(858) 707-4000

56977311

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC ACKNOWLEDGEMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/931,506	**05/04/2023 06:09:34 PM ET**	**RMMS.003C12**

Title of Invention

STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	3894	FILED BY	Tayler Amerine
PATENT CENTER #	62039585	FILING DATE	09/12/2022
CUSTOMER #	20995	FIRST NAMED INVENTOR	Andrey V. Kuzmin
CORRESPONDENCE ADDRESS	-	AUTHORIZED BY	Aaron Davis

Documents

TOTAL DOCUMENTS: 2

DOCUMENT	PAGES	DESCRIPTION	SIZE (KB)
RMMS_IFEE.pdf	1	Issue Fee Payment (PTO-85B)	185 KB
RMMS_COMMENTS.pdf	2	Miscellaneous Incoming Letter	21 KB

Digest

DOCUMENT	MESSAGE DIGEST(SHA-512)
RMMS_IFEE.pdf	FE26FD523BFAD45EA5832275504849E491359AFB3F2448DC387532B7EE580FF21BBE456117BA7CEB0596E5513143FFA416DF5F7F456A590D1A0C907A9DB86529

RMMS_COMMENTS.pdf EC47A0DCD9D5EB1441D1DD5959D6CEACB6C9F2167758BE8
0D7860A258F79C743C8D511990D552C71E931E626E7C7A7296
1E7F50C2EF0ECB32006BE2FA0E63191

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 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.

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC PAYMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/931,506	05/04/2023 06:09:34 PM ET	RMMS.003C12

Title of Invention

STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	3894	FILED BY	Tayler Amerine
PATENT CENTER #	62039585	AUTHORIZED BY	Aaron Davis
CUSTOMER #	20995	FILING DATE	09/12/2022
CORRESPONDENCE ADDRESS	-	FIRST NAMED INVENTOR	Andrey V. Kuzmin

Payment Information

PAYMENT METHOD	PAYMENT TRANSACTION ID	PAYMENT AUTHORIZED BY
CARD / 2598	E202354I10469508	Wendy Castorena

PRE-AUTHORIZED ACCOUNT	PRE-AUTHORIZED CATEGORY
111410	37 CFR 1.16 (National application filing, search, and examination fees); 37 CFR 1.17 (Patent application and reexamination processing fees)

FEE CODE	DESCRIPTION	ITEM PRICE($)	QUANTITY	ITEM TOTAL($)
2501	UTILITY ISSUE FEE	480.00	1	480.00
			TOTAL AMOUNT:	$480.00

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 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.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), by mail or fax, or via EFS-Web.

By mail, send to: Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

By fax, send to: (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 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. **Because electronic patent issuance may occur shortly after issue fee payment, any desired continuing application should preferably be filed prior to payment of this issue fee in order not to jeopardize copendency.**

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

20995 7590 04/12/2023
KNOBBE MARTENS OLSON & BEAR LLP
2040 MAIN STREET
FOURTEENTH FLOOR
IRVINE, CA 92614

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 transmitted to the USPTO via EFS-Web or by facsimile to (571) 273-2885, on the date below.

(Typed or printed name)
(Signature)
(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/931,506	09/12/2022	Andrey V. Kuzmin	RMMS.003C12	3894

TITLE OF INVENTION: STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$480	$0.00	$0.00	$480	07/12/2023

EXAMINER	ART UNIT	CLASS-SUBCLASS
BATAILLE, PIERRE MICHE	2136	710-301000

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/AIA/122 or PTO/SB/122) attached.

☐ "Fee Address" indication (or "Fee Address" Indication form PTO/AIA/47 or 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 Knobbe, Martens,
2 Olson & Bear, LLP
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 must have been previously recorded, or filed for recordation, as set forth in 37 CFR 3.11 and 37 CFR 3.81(a). Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE: Radian Memory Systems, Inc.

(B) RESIDENCE: (CITY and STATE OR COUNTRY): Manhattan Beach, California

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

4a. Fees submitted: ☒ Issue Fee ☐ Publication Fee (if required)

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

☒ Electronic Payment via Patent Center or EFS-Web ☐ Enclosed check ☐ Non-electronic payment by credit card (Attach form PTO-2038)

☒ The Director is hereby authorized to charge ~~the required fee(s),~~ any deficiency, or credit any overpayment to Deposit Account No. 11-1410

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

☐ Applicant certifying micro entity status. See 37 CFR 1.29

NOTE: Absent a valid certification of Micro Entity Status (see forms PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.

☐ Applicant asserting small entity status. See 37 CFR 1.27

NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.

☐ Applicant changing to regular undiscounted fee status.

NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: This form must be signed in accordance with 37 CFR 1.31 and 1.33. See 37 CFR 1.4 for signature requirements and certifications.

Authorized Signature /Aaron M. Davis/ Date May 4, 2023

Typed or printed name Aaron M. Davis Registration No. 62,027

SAMSUNG EX1002, Page 25 of 1031

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

20995 7590 04/12/2023

KNOBBE MARTENS OLSON & BEAR LLP
2040 MAIN STREET
FOURTEENTH FLOOR
IRVINE, CA 92614

EXAMINER
BATAILLE, PIERRE MICHE

ART UNIT	PAPER NUMBER
2136	

DATE MAILED: 04/12/2023

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/931,506	09/12/2022	Andrey V. Kuzmin	RMMS.003C12	3894

TITLE OF INVENTION: STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$480	$0.00	$0.00	$480	07/12/2023

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 ENTITY STATUS shown above. If the ENTITY STATUS is shown as SMALL or MICRO, verify whether entitlement to that entity status still applies.

If the ENTITY STATUS is the same as shown above, pay the TOTAL FEE(S) DUE shown above.

If the ENTITY STATUS is changed from that shown above, on PART B - FEE(S) TRANSMITTAL, complete section number 5 titled "Change in Entity Status (from status indicated above)".

For purposes of this notice, small entity fees are 40% the amount of undiscounted fees, and micro entity fees are 20% the amount of undiscounted fees.

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. 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: Maintenance fees are due in utility patents issuing on applications filed on or after Dec. 12, 1980. It is patentee's responsibility to ensure timely payment of maintenance fees when due. More information is available at www.uspto.gov/PatentMaintenanceFees.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), by mail or fax, or via EFS-Web.

By mail, send to: Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

By fax, send to: (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 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. **Because electronic patent issuance may occur shortly after issue fee payment, any desired continuing application should preferably be filed prior to payment of this issue fee in order not to jeopardize copendency.**

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

20995 7590 04/12/2023

KNOBBE MARTENS OLSON & BEAR LLP
2040 MAIN STREET
FOURTEENTH FLOOR
IRVINE, CA 92614

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 transmitted to the USPTO via EFS-Web or by facsimile to (571) 273-2885, on the date below.

(Typed or printed name)
(Signature)
(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/931,506	09/12/2022	Andrey V. Kuzmin	RMMS.003C12	3894

TITLE OF INVENTION: STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$480	$0.00	$0.00	$480	07/12/2023

EXAMINER	ART UNIT	CLASS-SUBCLASS
BATAILLE, PIERRE MICHE	2136	710-301000

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/AIA/122 or PTO/SB/122) attached.

❑ "Fee Address" indication (or "Fee Address" Indication form PTO/AIA/47 or 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 must have been previously recorded, or filed for recordation, as set forth in 37 CFR 3.11 and 37 CFR 3.81(a). 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. Fees submitted: ❑ Issue Fee ❑ Publication Fee (if required)

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

❑ Electronic Payment via Patent Center or EFS-Web ❑ Enclosed check ❑ Non-electronic payment by credit card (Attach form PTO-2038)

❑ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment to Deposit Account No. _____

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

❑ Applicant certifying micro entity status. See 37 CFR 1.29

NOTE: Absent a valid certification of Micro Entity Status (see forms PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.

❑ Applicant asserting small entity status. See 37 CFR 1.27

NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.

❑ Applicant changing to regular undiscounted fee status.

NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: This form must be signed in accordance with 37 CFR 1.31 and 1.33. See 37 CFR 1.4 for signature requirements and certifications.

Authorized Signature _____ Date _____

Typed or printed name _____ Registration No. _____

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.
17/931,506	09/12/2022	Andrey V. Kuzmin	RMMS.003C12	3894

20995 7590 04/12/2023

KNOBBE MARTENS OLSON & BEAR LLP
2040 MAIN STREET
FOURTEENTH FLOOR
IRVINE, CA 92614

EXAMINER	
BATAILLE, PIERRE MICHE	
ART UNIT	**PAPER NUMBER**
2136	

DATE MAILED: 04/12/2023

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

The Office has discontinued providing a Patent Term Adjustment (PTA) calculation with the Notice of Allowance.

Section 1(h)(2) of the AIA Technical Corrections Act amended 35 U.S.C. 154(b)(3)(B)(i) to eliminate the requirement that the Office provide a patent term adjustment determination with the notice of allowance. See Revisions to Patent Term Adjustment, 78 Fed. Reg. 19416, 19417 (Apr. 1, 2013). Therefore, the Office is no longer providing an initial patent term adjustment determination with the notice of allowance. The Office will continue to provide a patent term adjustment determination with the Issue Notification Letter that is mailed to applicant approximately three weeks prior to the issue date of the patent, and will include the patent term adjustment on the patent. Any request for reconsideration of the patent term adjustment determination (or reinstatement of patent term adjustment) should follow the process outlined in 37 CFR 1.705.

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.

OMB Clearance and PRA Burden Statement for PTOL-85 Part B

The Paperwork Reduction Act (PRA) of 1995 requires Federal agencies to obtain Office of Management and Budget approval before requesting most types of information from the public. When OMB approves an agency request to collect information from the public, OMB (i) provides a valid OMB Control Number and expiration date for the agency to display on the instrument that will be used to collect the information and (ii) requires the agency to inform the public about the OMB Control Number's legal significance in accordance with 5 CFR 1320.5(b).

The information collected by PTOL-85 Part B 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 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, 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.

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

SAMSUNG EX1002 Page 29 of 1031

Notice of Allowability	Application No. 17/931,506	Applicant(s) Kuzmin et al.	
	Examiner PIERRE MICHEL BATAILLE	Art Unit 2136	AIA (FITF) Status Yes

-- 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 0/23/2023.
   ☐ A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on _____.

2. ☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on _____; the restriction requirement and election have been incorporated into this action.

3. ☑ The allowed claim(s) is/are 2-31 . As a result of the allowed claim(s), you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see **http://www.uspto.gov/patents/init_events/pph/index.jsp** or send an inquiry to **PPHfeedback@uspto.gov.**

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

   Certified copies:

   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.

5. ☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
   ☐ 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. ☑ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date 03/23/2023.
3. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material _____.
4. ☐ Interview Summary (PTO-413), Paper No./Mail Date. _____.
5. ☐ Examiner's Amendment/Comment
6. ☐ Examiner's Statement of Reasons for Allowance
7. ☐ Other _____.

/PIERRE MICHEL BATAILLE/
Primary Examiner, Art Unit 2136

Notice of References Cited	Application/Control No. 17/931,506	Applicant(s)/Patent Under Reexamination Kuzmin et al.	
	Examiner PIERRE MICHEL BATAILLE	Art Unit 2136	Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	CPC Classification	US Classification
*	A	US-20120166911-A1	06-2012	Takahashi; Kyosuke	G06F11/1048	714/E11.054
*	B	US-20070258162-A1	11-2007	Nakagawa; Takahiro	G11B5/024	360/77.02
*	C	US-20050141313-A1	06-2005	Gorobets, Sergey Anatolievich	G06F11/141	711/E12.008
	D					
	E					
	F					
	G					
	H					
	I					
	J					
	K					
	L					
	M					

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	CPC Classification
	N	WO-2010027983-A1	03-2010	WO	GOTO A	G11C16/10
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/931,506	Kuzmin et al.
	Examiner	Art Unit
	PIERRE MICHEL BATAILLE	2136

CPC - Searched*		
Symbol	Date	Examiner
G06F12/0246, G06F12/10, G06F12/02, G06F3/0611, G06F3/0679, G06F3/0638	12/21/2022	PMB

CPC Combination Sets - Searched*		
Symbol	Date	Examiner

US Classification - Searched*			
Class	Subclass	Date	Examiner

* See search history printout included with this form or the SEARCH NOTES box below to determine the scope of the search.

Search Notes		
Search Notes	Date	Examiner
PALM INVENTOR & ASSIGNEE NAME SEARCH	11/16/2022	PMB
KEYORD & CONTEXT PE2E SEARCH	11/16/2022	PMB
NON-PATENT LITERATURE SEARCH	11/16/2022	PMB
KEYORD & CONTEXT PE2E SEARCH	11/17/2022	PMB
NON-PATENT LITERATURE SEARCH	12/21/2022	PMB
KEYORD & CONTEXT PE2E SEARCH	12/21/2022	PMB
KEYORD & CONTEXT PE2E SEARCH	04/08/2023	PMB
NON-PATENT LITERATURE SEARCH	04/08/2023	PMB

/PIERRE MICHEL BATAILLE/ Primary Examiner, Art Unit 2136	

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/931,506	Kuzmin et al.
	Examiner	Art Unit
	PIERRE MICHEL BATAILLE	2136

Interference Search			
US Class/CPC Symbol	**US Subclass/CPC Group**	**Date**	**Examiner**
	INTERFERENCE SEARCH INCLUDED IN WEST SEARCH	11/16/2022	PMB
	INTERFERENCE SEARCH INCLUDED IN WEST SEARCH	12/21/2022	PMB
	INTERFERENCE SEARCH INCLUDED IN WEST SEARCH	04/08/2023	PMB

/PIERRE MICHEL BATAILLE/ Primary Examiner, Art Unit 2136	

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/931,506	Kuzmin et al.
	Examiner	**Art Unit**
	PIERRE MICHEL BATAILLE	2136

CPC						
Symbol					**Type**	**Version**
G06F	/	12	/	0246	F	2013-01-01
G06F	/	16	/	1847	I	2019-01-01
G06F	/	11	/	1068	I	2013-01-01
G06F	/	3	/	0611	I	2013-01-01
G06F	/	12	/	10	I	2013-01-01
G06F	/	8	/	654	I	2018-02-01
G06F	/	3	/	0679	I	2013-01-01
G06F	/	3	/	0638	I	2013-01-01
G06F	/	3	/	0688	I	2013-01-01
G06F	/	11	/	1072	I	2013-01-01
G06F	/	2015	/	766	A	2013-01-01
G06F	/	2212	/	7208	A	2013-01-01
G06F	/	2212	/	2022	A	2013-01-01
G06F	/	2212	/	7207	A	2013-01-01

CPC Combination Sets								
Symbol					**Type**	**Set**	**Ranking**	**Version**
	/		/					

NONE		**Total Claims Allowed:**	
(Assistant Examiner)	(Date)	30	
/PIERRE MICHEL BATAILLE/ Primary Examiner, Art Unit 2136	08 April 2023	O.G. Print Claim(s)	O.G. Print Figure
(Primary Examiner)	(Date)	1	15

U.S. Patent and Trademark Office — Part of Paper No.: 20230408

SAMSUNG EX1002, Page 34 of 1031

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/931,506	Kuzmin et al.
	Examiner	**Art Unit**
	PIERRE MICHEL BATAILLE	2136

INTERNATIONAL CLASSIFICATION					
CLAIMED					
G06F	/	12	/	02	
G06F	/	12	/	10	

NON-CLAIMED				
	/		/	

US ORIGINAL CLASSIFICATION	
CLASS	**SUBCLASS**

CROSS REFERENCES(S)						
CLASS	**SUBCLASS (ONE SUBCLASS PER BLOCK)**					

NONE		Total Claims Allowed:	
(Assistant Examiner)	(Date)	30	
/PIERRE MICHEL BATAILLE/ Primary Examiner, Art Unit 2136	08 April 2023	O.G. Print Claim(s)	O.G. Print Figure
(Primary Examiner)	(Date)	1	15

U.S. Patent and Trademark Office — Part of Paper No.: 20230408

SAMSUNG EX1002, Page 35 of 1031

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/931,506	Kuzmin et al.
	Examiner	**Art Unit**
	PIERRE MICHEL BATAILLE	2136

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

CLAIMS

Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original
	1	9	10	18	19	27	28								
1	2	10	11	19	20	28	29								
2	3	11	12	20	21	29	30								
3	4	12	13	21	22	30	31								
4	5	13	14	22	23										
5	6	14	15	23	24										
6	7	15	16	24	25										
7	8	16	17	25	26										
8	9	17	18	26	27										

NONE		**Total Claims Allowed:**	
(Assistant Examiner)	(Date)	30	
/PIERRE MICHEL BATAILLE/ Primary Examiner, Art Unit 2136	08 April 2023	O.G. Print Claim(s)	O.G. Print Figure
(Primary Examiner)	(Date)	1	15

U.S. Patent and Trademark Office

Part of Paper No.: 20230408

SAMSUNG EX1002, Page 36 of 1031

PE2E SEARCH - Search History (Prior Art)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	British Equivalents	Time Stamp
L7	31	write near3 address WITH eras$ AND eras$ near4 (unit OR block) WITH (full OR empty) AND (full OR enpty) near3 (flag OR status OR state)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/11/17 07:25 AM
L8	7	write near3 address WITH eras$ AND eras$ near4 (unit OR block) WITH (full OR empty) AND (full OR enpty) near3 (flag OR status OR state)	(US-PGPUB; USPAT)	OR	ON	ON	2022/11/17 07:25 AM
L9	491	eras$ near4 (unit OR block) WITH (full OR empty) AND (full OR enpty) near3 (flag OR status OR state)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/11/17 07:27 AM
L10	188	L9 AND writ$ WITH eras$ near5 (block OR unit) WITH (state OR status)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/11/17 07:28 AM

L11	51	L10 AND eras$ WITH writ$ WITH available	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/11/17 07:30 AM
L12	221	eras$ near4 (unit OR block) WITH (full OR empty) AND (full OR enpty) near3 (flag OR status OR state)	(US-PGPUB; USPAT)	OR	ON	ON	2022/11/17 07:30 AM
L14	104	L12 AND writ$ WITH eras$ near5 (block OR unit) WITH (state OR status)	(US-PGPUB; USPAT)	OR	ON	ON	2022/11/17 07:30 AM
L15	59	L14 AND eras$ WITH writ$ WITH available	(US-PGPUB; USPAT)	OR	ON	ON	2022/11/17 07:31 AM
L19	11520	(G06F12/02 G06F12/0246G06F3/0611G06F3/0638G06F3/0679G06F3/0688G06F11/1068G06F11/1072).CPC.	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2023/04/08 01:48 PM
L20	4	L4 AND eras$ near3 (unit OR block) WITH (full OR empty) AND (full OR enpty) near3 state	(USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT;	OR	ON	ON	2023/04/08 01:48 PM

			IBM_TDB)				
L21	2	L4 AND eras$ near3 (unit OR block) WITH (full OR empty) AND (full OR enpty) near3 state	(US-PGPUB; USPAT)	OR	ON	ON	2023/04/08 01:48 PM
L22	31	write near3 address WITH eras$ AND eras$ near4 (unit OR block) WITH (full OR empty) AND (full OR enpty) near3 (flag OR status OR state)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2023/04/08 01:48 PM
L23	7	write near3 address WITH eras$ AND eras$ near4 (unit OR block) WITH (full OR empty) AND (full OR enpty) near3 (flag OR status OR state)	(US-PGPUB; USPAT)	OR	ON	ON	2023/04/08 01:48 PM
L24	521	eras$ near4 (unit OR block) WITH (full OR empty) AND (full OR enpty) near3 (flag OR status OR state)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2023/04/08 01:48 PM
L25	200	L9 AND writ$ WITH eras$ near5 (block OR unit) WITH (state OR status)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO;	OR	ON	ON	2023/04/08 01:48 PM

			JPO; DERWENT; IBM_TDB)				
L26	58	L10 AND eras$ WITH writ$ WITH available	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2023/04/08 01:48 PM
L27	238	eras$ near4 (unit OR block) WITH (full OR empty) AND (full OR enpty) near3 (flag OR status OR state)	(US-PGPUB; USPAT)	OR	ON	ON	2023/04/08 01:48 PM
L28	117	L12 AND writ$ WITH eras$ near5 (block OR unit) WITH (state OR status)	(US-PGPUB; USPAT)	OR	ON	ON	2023/04/08 01:48 PM
L29	72	L14 AND eras$ WITH writ$ WITH available	(US-PGPUB; USPAT)	OR	ON	ON	2023/04/08 01:48 PM
L30	0	L29 AND (multiplane OR muti?plane) near4 (write OR request$) AND write near3 (request r OR command)	(US-PGPUB; USPAT)	OR	ON	ON	2023/04/08 01:54 PM
L31	26	(multiplane OR muti?plane) near4 (write OR request$) AND write near3 (request r OR command)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2023/04/08 01:55 PM
L32	11	(multiplane OR muti?plane) near4 (write OR request$) AND write near3 (request r OR command)	(US-PGPUB; USPAT)	OR	ON	ON	2023/04/08 01:55 PM

L33	0	(multiplane OR muti?plane) near4 (write OR request$) NEAR5 command WITH (zone OR area)	(US-PGPUB; USPAT)	OR	ON	ON	2023/04/08 03:11 PM
L34	0	(multiplane OR muti?plane) near4 (write OR request$) NEAR5 command WITH (zone OR area)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2023/04/08 03:11 PM
L35	2	(multi$storage OR muti?storage) near4 (write OR request$) NEAR5 command WITH (storage OR memory)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2023/04/08 03:14 PM
L36	0	(multi$storage OR muti?storage) near4 (write OR request$) NEAR5 command WITH (storage OR memory)	(US-PGPUB; USPAT)	OR	ON	ON	2023/04/08 03:14 PM
L37	24	("20050141313" "20090019217" "7913061" "20070258162" "7532426" "20030103385" "20080091901" "20080091871" "20110202578" "8504798" "20070113030" "20060155921" "7624239" "20070113029" "20060155920"	(US-PGPUB; USPAT)	OR	ON	ON	2023/04/08 03:59 PM

		"8392476" "20090216938" "7412560" "7783845" "20130031317" "20050144358" "20050141312" "7366826" "20090019218").pn.					

PE2E SEARCH - Search History (Interference)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	British Equivalents	Time Stamp
N1	2	(write near3 address WITH eras$ AND eras$ near4 (unit OR block) WITH (full OR empty) AND (full OR enpty) near3 (flag OR status OR state)).clm.	(US-PGPUB; USPAT)	OR	ON	ON	2022/11/17 07:26 AM
N2	1	((multiplane OR muti?plane) near4 (write OR request$) AND write near3 (request OR command)).CLM.	(US-PGPUB; USPAT)	OR	ON	ON	2023/04/08 02:40 PM

Other Applications of Assignee

Applicant wishes to draw the Examiner's attention to, and encourages the Examiner to review, the following co-owned patents and/or applications and their existing and ongoing prosecution history:

Application No.	Filing Date	Patent No.
13/767,723	2/14/2013	9,652,376
14/047,193	10/7/2013	9,229,854
14/951,708	11/25/2015	9,519,578
15/346,641	11/8/2016	9,710,377
15/621,888	6/13/2017	10,445,229
16/570,922	9/13/2019	11,118,457
16/751,925	1/24/2020	11,314,636
16/783,083	2/5/2020	11,334,479
16/794,805	2/19/2020	
16/833,547	3/28/2020	11,216,365
17/563,481	12/28/2021	11,487,657
17/563,467	12/28/2021	11,487,656
17/571,394	1/7/2022	
17/877,881	7/29/2022	
17/931,516	9/12/2022	
17/952,964	9/23/2022	
17/952,230	9/24/2022	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /P.B/

SAMSUNG EX1002, Page 43 of 1031

Applicant notes that cited references, office actions, responses, and notices of allowance currently exist or will exist with reference to the above-referenced matters. Applicant also understands that the Examiner has access to sophisticated Patent Office computing systems that provide ready access to the full file histories of these matters including, for example, specifications, drawings, pending claims, cited art, office actions, responses, declarations, and notices of allowance. Rather than submit copies of these file histories, Applicant respectfully requests that the Examiner continue to review these file histories online for past, current, and future information about these matters that may be relevant to examination of the present application. With respect to parent applications of the present application, Applicant understands that, in accordance with MPEP 609, the Examiner of the present application will consider information which has been considered by the Patent Office in the parent applications. If the Examiner cannot readily access these file histories and information, Applicant would be pleased to provide any portion of any of the file histories or information at any time upon specific Examiner request.

Timing of Disclosure

This Information Disclosure Statement is being filed with a Request for Continued Examination, and no additional fee is believed to be required. The Commissioner is hereby authorized to charge any additional fees which may be required or to credit any overpayment to Account No. 11-1410.

Respectfully submitted,

KNOBBE, MARTENS, OLSON & BEAR, LLP

Dated: March 23, 2023

By: /Aaron M. Davis/
Aaron M. Davis
Registration No. 62,027
Registered Practitioner
(858) 707-4000

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /P.B/

SAMSUNG EX1002, Page 44 of 1031

INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Application No.	17/931506
	Filing Date	September 12, 2022
	First Named Inventor	Kuzmin, Andrey V.
	Art Unit	2136
(Multiple sheets used when necessary)	Examiner	Bataille, Pierre Miche
SHEET 1 OF 1	Attorney Docket No.	RMMS.003C12

U.S. PATENT DOCUMENTS

Examiner Initials	Cite No.	Document Number *Number - Kind Code (if known)* Example: 1,234,567 B1	Publication Date MM-DD-YYYY	Name	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear
	1	9,489,296	11-08-2016	Tomlin et al.	
	2	11,416,413	08-16-2022	Lercari et al.	
	3	11,449,436	09-20-2022	Lercari et al.	
	4	11,537,528	12-27-2022	Lercari et al.	
	5	11,537,529	12-27-2022	Lercari et al.	
	6	11,544,200	01-03-2023	Lercari et al.	
	7	2004/0083335	04-29-2004	Gonzalez	
	8	2005/0141313	06-30-2005	Gorobets et al.	
	9	2010/0026773	02-04-2010	Schultz et al.	
	10	2010/0077136	03-25-2010	Ware et al.	
	11	2010/0250837	09-30-2010	Schmidberger et al.	
	12	2011/0082967	04-07-2011	Deshkar et al.	
	13	2011/0161560	06-30-2011	Hutchison et al.	
	14	2012/0260150	10-11-2012	Cideciyan et al.	
	15	2013/0019057	01-17-2013	Stephens	
	16	2013/0024460	01-24-2013	Peterson et al.	
	17	2013/0122781	05-16-2013	Rutten et al.	

FOREIGN PATENT DOCUMENTS

Examiner Initials	Cite No.	Foreign Patent Document *Country Code-Number-Kind Code* Example: JP 1234567 A1	Publication Date MM-DD-YYYY	Name	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear	T[1]
	18	WO 2011044154 A1	04-14-2011	Deshkar et al.		
	19	WO 2011055407 A1	05-12-2011	Takada et al.		

NON PATENT LITERATURE DOCUMENTS

Examiner Initials	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	T[1]

Examiner Signature	/PIERRE MICHEL BATAILLE/	Date Considered	04/08/2023

***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 next communication to applicant.

T[1] - Place a check mark in this area when an English language Translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /P.B/

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
11 March 2010 (11.03.2010)

PCT

(10) International Publication Number
WO 2010/027983 A1

(51) International Patent Classification:
G11C 7/10 (2006.01) *G06F 13/16* (2006.01)
G11C 16/10 (2006.01)

(21) International Application Number:
PCT/US2009/055632

(22) International Filing Date:
1 September 2009 (01.09.2009)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/093,872 3 September 2008 (03.09.2008) US

(71) Applicant *(for all designated States except US)*: **MARVELL WORLD TRADE LTD.**; L 'Horizon, Gunsite Road, Brittons Hill, St. Michael, Barbados BB 14027 (BB).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **GOTO, Akio** [JP/JP]; No. 1010-1-703, Tsuji Hatogayashi, Saitama 334-0004 (JP). **LEE, Chi Kong** [CA/US]; 43175 Stan- Street, Fremont, California 94539 (US). URABE, **Masayuki** [JP/JP]; 1723 Ishida, Isehara City, Kanagawa 259-1 116 (JP).

(74) Agent: HUNTER, William, E.; Fish & Richardson P.C., P. O. Box 1022, Minneapolis, Minnesota 55440-1022 (US).

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

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM,

[Continued on next page]

(54) Title: PROGAMMING DATA INTO A MULTI-PLANE FLASH MEMORY

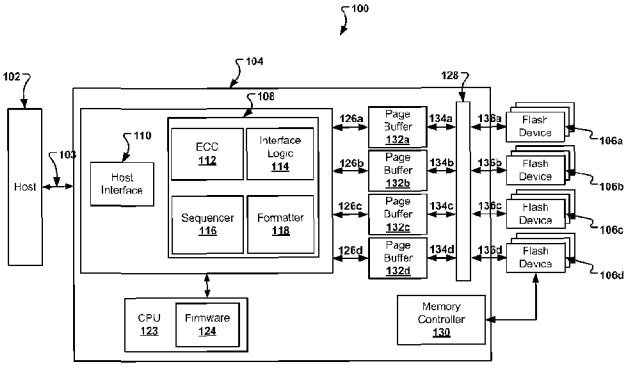


FIG. 1A

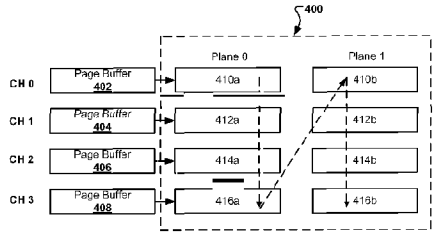


FIG. 4A

(57) Abstract: Systems, methods and computer program products for programming data into a multi-plane memory device employ a multi-plane data order. To allow multiple data pages to be programmed without a need to increase the size of page buffers, in some implementations, a data transfer scheme at which the data pages are programmed can be manipulated. Specifically, data across all channels can first be programmed into a first plane of the multi-plane flash memory device in parallel. While the data transfer program operation is in progress, data to be programmed into a succeeding plane (e.g., plane "1") can be read into and cached in one or more page buffers. After the data transfer program for the first plane is complete, data cached in the page buffers can be immediately latched and programmed into the multi-plane flash memory device.

WO 2010/027983 A1

TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(H))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(Hi))*

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

PROGAMMING DATA INTO A MULTI-PLANE FLASH MEMORY

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority to U.S. Provisional Application Serial No. 61/093,872 titled "MULTI-PLANE DATA ORDER," filed on September 3, 2008, the disclosure of which is incorporated herein by reference in its entirety.

TECHNICAL FIELD

[0002] The subject matter of this application is generally related to memory devices.

BACKGROUND

[0003] Flash memory is a common type of non-volatile memory used for storage in consumer electronics. Flash memory is widely used for storing application data and/or media data in popular consumer products such as digital audio/video players, cell phones and digital cameras. Flash memory retains stored data in the absence of power, which provides a power savings advantage for consumer products. Flash memory can further be used as a dedicated storage device, such as a portable flash drive pluggable into a universal serial port (USB) of a personal computer, and a magnetic hard disk drive (HDD) replacement.

[0004] Recently, manufacturers have introduced flash memory devices with configurations that include multiple memory arrays, or memory planes. Multi-plane memory devices have been implemented in single-level cell memory devices and multi-level cell memory devices. However, multi-plane memory devices often have low read, program and erase speed. The speed of multi-plane memory devices can become a critical performance issue as the number of memory planes increases. While the multi-plane configuration allows data from one channel to be transferred to a buffer during a wait READ/PROGRAM interval to improve the data transfer time associated with this one channel, the multi-plane configuration can cause reduced system performance when data is transferred from multiple channels.

SUMMARY

[0005] Systems, methods and computer program products for programming data into a multi-plane memory device employ a multi-plane data order. To allow multiple data pages to be programmed without a need to increase the size of page buffers, in some implementations, a data transfer scheme at which the data pages are programmed can be

SAMSUNG EX1002, Page 48 of 1031

manipulated. Specifically, data across all channels can first be programmed into a first plane of the multi-plane flash memory device in parallel. While the data transfer program operation is in progress, data to be programmed into a succeeding plane (e.g., plane "1") can be read into and cached in one or more page buffers. After the data transfer program for the first plane is complete, data cached in the page buffers can be immediately latched and programmed into the multi-plane flash memory device.

[0006] In some implementations, a method can be used that includes receiving first data and second data, storing the first data in a first page buffer and the second data in a second page buffer, programming the stored first data into a first page associated with a memory plane of a multi-plane memory device, and programming the stored second data into a second page associated with the memory plane in parallel with programming the stored first data.

[0007] In some implementations, the first data can include a first subset of the first data and a second subset of the first data, and the second data can include a first subset of the second data and a second subset of the second data.

[0008] In some implementations, the first subset of the first data can be programmed into the first page associated with the memory plane, and the first subset of the second data can be programmed into the second page associated with the memory plane.

[0009] In some implementations, the memory plane can be a first memory plane, the second subset of the first data can be programmed into a first page associated with a second memory plane of the multi-plane memory device, and the second subset of the second data can be programmed into the second page associated with the second memory plane.

[0010] In some implementations, programming the second subset of the first data into the first page associated with the second memory plane can be performed after programming the first subset of the second data into the second page associated with the first memory plane.

[0011] In some implementations, programming the second subset of the second data into the second page associated with the second memory plane can be performed after programming the second subset of the first data into the first page associated with the second memory plane.

[0012] In some implementations, a device can be used that includes a first page buffer configured to receive first data from a first channel and store the first data, a second page buffer configured to receive second data from a second channel and store the second data, and circuitry configured to program the stored first data into a first page associated with a

SAMSUNG EX1002, Page 49 of 1031

memory plane and the stored second data into a second page associated with the memory plane in parallel.

[0013] In some implementations, the first data can include a first subset of the first data and a second subset of the first data, and the second data can include a first subset of the second data and a second subset of the second data.

[0014] In some implementations, the circuitry can be configured to program the first subset of the first data into the first page associated with the memory plane, and the first subset of the second data into the second page associated with the memory plane.

[0015] In some implementations, the memory plane can be a first memory plane, and the circuitry can be configured to program the second subset of the first data into a first page associated with a second memory plane, and the second subset of the second data into the second page associated with the second memory plane.

[0016] In some implementations, the circuitry can be configured to program the second subset of the first data into the first page associated with the second memory plane after the first subset of the second data is programmed into the second page associated with the first memory plane.

[0017] In some implementations, the circuitry can be configured to program the second subset of the second data into the second page associated with the second memory plane after the second subset of the first data is programmed into the first page associated with the second memory plane.

[0018] In some implementations, a system can be used that includes a memory array having a memory plane, a first page buffer to receive first data from a first channel and store the first data, a second page buffer to receive second data from a second channel and store the second data, and a controller can be configured to program the stored first data into a first page associated with the memory plane and the stored second data into a second page associated with the memory plane in parallel.

[0019] In some implementations, the controller can be configured to initialize a program operation associated with the stored second data before completing the programming of the stored first data.

[0020] In some implementations, the memory plane can be a first memory plane, the memory array can include a second memory plane, the first data can include a first subset and a second subset, and the controller can be configured to program the first subset into the first page associated with the first memory plane and the second subset into a first page associated with the second memory plane.

SAMSUNG EX1002, Page 50 of 1031

[0021] In some implementations, the second data can include a first subset and a second subset, and the controller can be configured to program the first subset of the second data into the second page associated with the first memory plane and the second subset of the second data into a second page associated with the second memory plane.

[0022] In some implementations, the controller can be configured to initialize a program operation associated with the first subset of the second data before initializing a program operation associated with the second subset of the first data but after initializing a program operation associated with the first subset of the first data.

[0023] The details of one or more embodiments of the invention are set forth in the accompanying drawings and the description below. Other features, objects, and advantages of the invention will be apparent from the description and drawings, and from the claims.

DESCRIPTION OF DRAWINGS

[0024] FIG. 1A shows an example of a solid state drive system.

[0025] FIG. 1B shows an example of a multi-plane flash memory device.

[0026] FIG. 2A shows an example of a timing diagram associated with each pin in a NAND flash interface for a READ operation.

[0027] FIG. 2B shows an example of a timing diagram associated with each pin in a NAND flash interface for a PROGRAM operation.

[0028] FIG. 2C shows an example of a timing diagram associated with each pin in a NAND flash interface for an ERASE operation.

[0029] FIG. 3A shows an example of a horizontal data transfer scheme for a two-plane flash memory device.

[0030] FIG. 3B shows an example of a timing diagram associated with the horizontal data transfer scheme shown in FIG. 3A.

[0031] FIG. 4A shows an example of a vertical data transfer scheme.

[0032] FIG. 4B shows an example of a timing diagram associated with the vertical data transfer scheme shown in FIG. 4A.

[0033] FIG. 5 shows an example of a process for executing a vertical data transfer scheme.

[0034] Like reference symbols in the various drawings indicate like elements.

SAMSUNG EX1002, Page 51 of 1031

DETAILED DESCRIPTION

[0035] Non-volatile solid-state memory, particularly electrically erasable and programmable read-only memory (EEPROM) and flash EEPROM packaged as a small form factor card, can be electrically erased and new data can be programmed or re-programmed. Generally, operations of a EEPROM memory can include a PROGRAM mode for writing data into a memory cell, a READ mode for reading out the data stored in the memory cell, and an ERASE mode for deleting the stored data.

[0036] In a flash memory device, ERASE operation can be performed per memory block or sector, and PROGRAM operation can be performed per page corresponding to a plurality of memory cells commonly coupled to a word line. The flash memory device can be configured as a NAND flash memory device in which cell transistors are coupled in parallel between a bitline and a ground electrode.

[0037] A flash memory device can be structured as a multi-plane memory device. Specifically, the multi-plane memory device can include a plurality of memory planes (e.g., a memory plane can include a memory array with its corresponding row decoder, page buffer and column decoder). Each memory plane can include one or more memory blocks arranged in a column direction and each block can correspond to a memory plane.

[0038] A multi-plane flash memory device can support multi-plane PROGRAM or READ commands, which program data into or read data from multiple pages associated with multiple different planes. For example, a multi-plane flash memory device can support multi-plane commands that write eight data pages into eight respective memory planes. The use of multiple planes and multi-plane PROGRAM commands allows a multi-plane memory device to perform multiple PROGRAM operations in parallel and enhance the PROGRAM speed of the multi-plane memory device.

[0039] To program data received over a series of channels into respective pages of a multi-plane memory device, the data can be loaded and programmed into a respective memory plane one at a time. For example, after programming data from a first channel into the page of a plane, data from a second channel can be loaded and programmed into the multi-plane memory device using the same procedure. This process, however, can prolong the PROGRAM operation time because the process does not permit parallel processing across the channels. Specifically, this process allows only one data page to be programmed at a time, and PROGRAM operations for other channels can commence only after the PROGRAM operation for a preceding channel has ended. These limitations prolong the

SAMSUNG EX1002, Page 52 of 1031

PROGRAM operation, and can adversely impact the speed performance of a multi-plane memory device.

[0040] Further, a multi-plane memory device typically includes page buffers to temporarily store pending data. As the number of data pages to be programmed increases, so does the need to increase the size of these page buffers to accommodate the increasing load of data. Increasing the size of these page buffers, however, can increase the cost of the page buffers and the complexity of the multi-plane memory device, and drive up the overall manufacturing cost to produce such a device.

[0041] To allow multiple data pages to be programmed without the need to increase the size of the page buffers, in some implementations, a vertical data transfer scheme specifying a vertical data transfer order can be used in PROGRAM operations. Specifically, data across all channels can first be programmed into pages associated with a first plane (e.g., plane "0") of the multi-plane flash memory device in parallel. While the PROGRAM operation is in progress, data to be programmed into pages associated with a succeeding plane (e.g., plane "1") can be read into and cached in one or more page buffers. Upon completing the PROGRAM operation for the first plane, data for the succeeding plane can be loaded from the page buffers and latched and programmed into the multi-plane memory device. This "vertical" data transfer scheme allows each plane (as opposed to each channel) to be "vertically" programmed independently of other planes such that the size of a page buffer associated with a particular channel need only accommodate one page instead of multiple pages of multiple planes, as will be discussed in greater detail below.

[0042] FIG. IA shows an example of a solid state drive system 100. As shown in FIG. IA, the system 100 includes a host 102 and a solid state drive 104. The solid state drive 104 can include a host interface 110, a central processor unit (CPU) 123, a memory interface 128, a memory controller 130 and flash memory devices 106a, 106b, 106c and 106d.

[0043] The host 102 can communicate with the solid state drive 104 through the host interface 110. The host interface 110, in some implementations, can include a Serial Advanced Technology Attachment (SATA) interface or a Parallel Advanced Technology Attachment (PATA) interface. A SATA interface or PATA interface can be used to convert serial or parallel data into parallel or serial data, respectively. For example, if the host interface 110 includes a SATA interface, then the SATA interface can receive serial data transferred from the host 102 through a bus 103 (e.g., a SATA bus), and convert the received serial data into parallel data. In other implementations, the host interface 110 can include a

SAMSUNG EX1002, Page 53 of 1031

hybrid interface. In these implementations, the hybrid interface can be used in conjunction with, for example, a serial interface.

[0044] The host interface 110, in some implementations, can include one or more registers in which operating commands and addresses from the host 102 can be temporarily stored. The host interface 110 can communicate a PROGRAM or READ command to a solid state controller 108 in response to the stored information in the register(s).

[0045] In some implementations, the solid state drive 104 can include one or more channels 126a, 126b, 126c and 126d, and each channel 126a-126d can be configured to receive one or more control signals (e.g., four chip-enable signals) or READ, PROGRAM or ERASE data or commands from the host 102 or from the flash memories 106a-106d.

[0046] The solid state controller 108 can be configured to handle any command, status, or control request for access to the flash memory devices 106a-106d. For example, the solid state controller 108 can be configured to manage and control storage and retrieval of data in the flash memory devices 106a-106d. To initialize a READ, PROGRAM or ERASE operation, the solid state controller 108 can receive one or more service requests or commands (e.g., READ, PROGRAM and ERASE requests) from the host 102 (or from the memory controller 130).

[0047] In some implementations, the solid state controller 108 can be a part of a microcomputer system under the control of a microprocessor (not shown). The solid state controller 108 can control the flow of commands and data between the host 102 and the solid state drive 104. In some implementations, the solid state controller 108 can include read-only memory (ROM), random-access memory (RAM) and other internal circuits. The solid state controller 108, in some implementations, can be configured to support various functions associated with the flash memory devices 106a-106d, such as, without limitation, diagnosing the flash memory devices 106a-106d, sending commands (e.g., activation, READ, PROGRAM, ERASE, pre-charge and refresh commands) to the flash memory devices 106a-106d, and receiving status from the flash memory devices 106a-106d. The solid state controller 108 can be formed on a same or different chip as the flash memory devices 106a-106d or on a same chip. The solid state controller 108 also can be formed on a same or different chip as the solid state drive 104.

[0048] The flash memory devices 106a-106d can be coupled with the memory interface 128. In some implementations, if the flash memory devices 106a-106d include NAND-type memory devices, the memory interface 128 can be a NAND flash input/output interface.

SAMSUNG EX1002, Page 54 of 1031

[0049] As shown in FIG. IA, the solid state controller 108 can include an error checking code (ECC) module 112, interface logic 114, a sequencer 116 and a formatter 118. In some implementations, the solid state controller 108 can be coupled with the CPU 123 including embedded firmware 124 by which the solid state controller 108 can be controlled. The CPU 123 can include a microprocessor, a signal processor (e.g., a digital signal processor) or microcontroller. In some implementations, the CPU 123 with the embedded firmware 124 can reside outside of the solid state drive 104.

[0050] In some implementations, the solid state drive 104 (and/or the host 102) can be mounted on a system on-chip (SOC). The SOC, in these implementations, can be fabricated using, for example, a digital process. The SOC can include an embedded process system (e.g., an embedded CPU) separate from the CPU 123 in the solid state drive 104. The SOC also can include a SRAM (Static Random Access Memory), system logic, cache memory and cache controller for processing program code and data. The program code and data associated with the embedded process system can be stored in the flash memory devices 106a-106d, and communicated to the SOC through, for example, an SOC interface (not shown). The SOC interface can be used by a translator for translating information flowing between the interface and the internal bus structure of the SOC. Control signals can flow from the SOC to the flash memory devices 106a-106d while instructions and data can flow from the flash memory devices 106a-106d to the SOC during READ operations. Instructions and data also can be sent to the flash memory devices 106a-106d during PROGRAM operations.

[0051] In some implementations, the flash memory devices 106a-106d can be controlled by the memory controller 130. The host 102 can communicate with the flash memory devices 106a-106d through the memory controller 130. The memory controller 130 can be connected to the flash memory devices 106a-106d through a corresponding pin or terminal. In these implementations, the memory controller 130 can be implemented as an application specific integrated circuit (ASIC) or as an SOC.

[0052] Alternatively, the host 102 can communicate with the flash memory devices 106a-106d through the solid state controller 108. For example, the host 102 can transmit one or more commands to the solid state controller 108, and through the memory interface 128, the solid state controller 108 can send the host's commands to the flash memory devices 106a-106d. The memory interface 128 can be a NAND flash I/O interface. The NAND flash interface can include one or more pins each corresponding to a specific function, as will be discussed in greater detail below.

SAMSUNG EX1002, Page 55 of 1031

[0053] In some implementations, the flash memory devices 106a-106d can be coupled with a plurality of page buffers 132a, 132b, 132c and 132d. In some implementations, each flash memory device 106a-106d can be associated with a respective page buffer 132a-132d. For example, as illustrated in FIG. IA, the flash memory device 106a can be associated with the page buffer 132a; the flash memory device 106b can be associated with the page buffer 132b; the flash memory device 106c can be associated with the page buffer 132c; and the flash memory device 106d can be associated with the page buffer 132d. If desired, each flash memory device 106a-106d also can be associated with more than one page buffer.

[0054] As will be described in greater detail below, the page buffers 132a-132d can be used as a temporary storage during PROGRAM/READ operations on the memory cells of the selected page. Each page buffer 132a-132d can have a storage capability at least equal to the storage capability of a memory page (e.g., 2kB).

[0055] Each page buffer 132a-132d can store one page of write data to be programmed into one page of flash memory cells. Each page buffer 132a-132d can include, without limitation, registers, sense circuits for sensing data read from one page of flash memory cells, verify logic, latching circuits or writing drivers. During PROGRAM operations, each page buffer 132a-132d can latch PROGRAM data received over a respective channel 126a-126d, and transfer the latched data to the respective flash memory device 106a-106d. Each page buffer 132a-132d also can perform program verify operations to ensure that the programmed data has been properly programmed into the flash memory devices 106a-106d.

[0056] As discussed above, the page buffers 132a-132d can be connected to the solid state controller 108 through respective channels 126a-126d. In some implementations, the page buffers 132a-132d also can be connected to the memory interface 128 through respective channels 134a, 134b, 134c and 134d, and to the flash memory devices 106a-106d through channels 136a, 136b, 136c and 136d. In some implementations, both the memory interface 128 and the page buffers 132a-132d can be formed as internal components of the solid state drive 104 or the solid state controller 108. In other implementations, the page buffers 132a-132d can be formed as external components to the solid state drive 104 or the solid state controller 108.

[0057] To select a particular page buffer 132a-132d for storing PROGRAM data, input (e.g., user input or input provided by software programs) can be received through a double data rate (DDR) interface (not shown), which can be error-checked by the ECC

SAMSUNG EX1002, Page 56 of 1031

module 112. For READ data, the ECC module 112 can be used to correct any error present in the READ data retrieved from the flash memory devices 106a-106d. For PROGRAM data, the ECC module 112 can be used to add one or more redundant bits to the PROGRAM data. Once the PROGRAM data is written, the redundant bits allow error-correction to be performed in a subsequent READ cycle of the written PROGRAM data. The ECC module 112 can first select a set of data to be rewritten to the flash memory devices 106a-106d, add redundant data to the selected set of data, and pass the selected set of data with the redundant information to the flash memory devices 106a-106d through a corresponding channel 126a-126d (e.g., following a firmware request order issued by the firmware 124). Output of the ECC module 112 then can be used as a SELECT signal to designate a page buffer 132a-132d for a particular PROGRAM operation.

[0058] In some implementations, the flash memory devices 106a-106d can be multi-plane memory devices each including, for example, four planes. FIG. IB shows an example of a multi-plane flash memory device 138. Referring to FIG. IB, the multi-plane memory device 138 can include four memory planes; namely, plane "0", plane "1", plane "2" and plane "3". Although four planes are shown, the number of planes can vary depending on the application, and a greater or lesser number of planes than those shown also are contemplated.

[0059] Each of plane "0", plane "1", plane "2" and plane "3" can include one or more memory cell blocks. Each memory cell block can include one or more pages (e.g., each page being controlled by one or more word lines). As shown, plane "0" can include a memory cell block 140 with page 148 as the initial page of the memory cell block 140; plane "1" can include a memory cell block 142 with page 150 as the initial page of the memory cell block 142; plane "2" can include a memory cell block 144 with page 152 as the initial page of the memory cell block 144; and plane "3" can include a memory cell block 146 with page 154 as the initial page of the memory cell block 146.

[0060] In some implementations, each plane can be associated with an internal buffer (e.g., buffers internal to the flash memory devices). For example, plane "0" can be associated with internal buffer 162a; plane "1" can be associated with internal buffer 162b; plane "2" can be associated with internal buffer 162c; and plane "3" can be associated with internal buffer 162d. Although only four internal buffers 162a-162d are shown, one skilled in the art would readily recognize that each of plane "0", plane "1", plane "2" and plane "3" can be associated with more than one internal buffer, or as many internal buffers as bit line pairs.

[0061] Data can be loaded sequentially onto each of the internal buffers 162a-162d associated with each of plane "0", plane "1", plane "2" and plane "3". A vertical data

SAMSUNG EX1002, Page 57 of 1031

transfer scheme can be, for example, executed by the solid state controller 108 and used to program the respective pages of plane "0", plane "1", plane "2" and plane "3". Specifically, data received over a first channel "CH 0" (e.g., over channel 126a) can be loaded into internal buffer 162a, and programmed into the first page 140 of plane "0". While the PROGRAM operation for the first channel "CH 0" is in progress, data received over a second channel "CH 1" (e.g., over channel 126b) can be loaded into internal buffer 162a and programmed into a second page 156 of plane "0". Similarly, while the PROGRAM operations for the first channel "CH 0" and the second channel "CH 1" are in progress, data received over a third channel "CH 2" (e.g., over channel 126c) can be loaded into the internal buffer 162a and programmed into a third page 158 of plane "0". The foregoing PROGRAM process can repeat until data received from all channels (e.g., up to the last channel "CH 7") have been programmed into the respective page of plane "0" (e.g., page 160).

[0062] Each flash memory device 106a-106d can include a non-volatile (or volatile) memory (e.g., a single-level flash memory or a multi-level flash memory). In some implementations, the nonvolatile memory can include a NAND-type flash memory module. A NAND-type flash memory module can include a command/address/data multiplexed interface such that commands, data, and addresses can be provided through corresponding input/output pins. Advantages of using NAND-type flash memory devices as the flash memory devices 106a-106d include, without limitation, faster boot and resume times; longer battery life; and higher data reliability.

[0063] Each flash memory device 106a-106d can be connected to a respective channel 126a-126d. Each channel 126a-126d can support, for example, one or more input and output lines, chip select signal lines, chip enable signal lines and the like. The channel also can support other signal lines such as, without limitation, write enable, read enable, read/busy output, and reset signal lines. To increase the degree of parallelism, each flash memory device 126a-126d can have its own channel connected to the solid state drive 104 as shown in FIG. IA. For example, flash memory device 106a can be connected to the solid state drive 104 using channel 126a; flash memory device 106b can be connected to the solid state drive 104 using channel 126b; flash memory device 106c can be connected to the solid state drive 104 using channel 126c; and flash memory device 106d can be connected to the solid state drive 104 using channel 126d. In other implementations, the flash memory devices 106a-106d can share a common channel.

[0064] The flash memory devices 106a-106d can be connected to the solid state drive 104 using standard connectors. Examples of standard connectors can include, without

SAMSUNG EX1002, Page 58 of 1031

limitation, SATA, USB (Universal Serial Bus), SCSI (Small Computer System Interface), PCMCIA (Personal Computer Memory Card International Association), and IEEE-1394 (Firewire). In some implementations, the flash memory devices 106a-106d can be detachable using the standard connectors.

[0065] Each flash memory device 106a-106d can include one or more solid state storage elements (or cells) arranged in a bank. A solid state storage element can be partitioned into pages. In some implementations, a solid state storage element can have a capacity of 2000 bytes, which can correspond to one page. A solid state storage element, in some implementations, can provide a total capacity of 4000 bytes (i.e., 4kB). Each flash memory device 106a-106d also can include one or more banks each being selected using a chip enable signal or chip select signal. The chip enable or chip select signal can select one or more solid state storage elements in response to a host command.

[0066] In some implementations, each solid state storage element can include one or more single-level cell ("SLC") devices and/or multi-level cell ("MLC") devices. Specifically, each storage element in the flash memory device 106a-106d can be programmed as a single bit per cell or multiple bits per cell. Each cell's threshold voltage generally determines the type of data that is stored in the cell. For example, in a SLC memory device, a threshold voltage of 0.5V can indicate a programmed cell (e.g., logical "0" state) while a threshold voltage of -0.5V can indicate an erased cell (e.g., logical "1" state). The SLC or MLC devices can be selected using a chip enable signal or chip select signal, which can be generated by the solid state controller 108 using a combination of control and address information received from the host 102.

[0067] Where multiple banks of solid state storage elements are used, in some implementations, the solid state drive 104 can access more than one bank of solid state storage elements in a same flash memory device at the same time. The solid state drive 104 also can access different banks in different flash memory devices at the same time. The capability to access more than one bank allows the solid state drive 104 to fully utilize the available resources and channels 126a-126d to increase the overall performance of the solid state drive 104. Further, where the flash memory devices 106a-106d share a same memory input/output line and control signal (e.g., chip enable signal), the number of pins of the solid state controller 108 can be reduced to further minimize the cost for manufacturing the solid state drive 104.

[0068] A flash memory array of a flash memory device 106a-106d can typically be organized in bits. For example, the memory array can include 8-bit depth. The flash memory

SAMSUNG EX1002, Page 59 of 1031

array also can be organized in bytes. For example, the flash memory array can include a portion containing 2k bytes (or 2kB), and a portion containing 64 bytes. The memory array further can be organized into pages. For example, the flash memory array can include 512K pages. A single page can be organized into two portions: a first portion (e.g., portion representing 2kB), and a second portion (e.g., a portion representing 64 bytes). The second portion can generally correspond to an eight-bit wide data input/output (I/O) path (e.g., I/O [0] - I/O [7]). Even further, the flash memory array can be arranged in blocks. For example, the flash memory array can include a block which equates to 64 pages. In all, a flash memory device can be formed using the foregoing bits, bytes, pages and blocks.

[0069] Each of the flash memory devices 106a-106d can be a multi-plane NAND flash memory device. NAND flash memory devices generally have faster PROGRAM, READ and ERASE times, higher density, lower cost per bit, and more endurance than NOR-type flash memory devices. A NAND flash memory can be coupled with a NAND flash I/O interface that allows sequential access to data. A NAND flash I/O interface can include multiple pins each corresponding to a specific function. An example interface is shown in TABLE 1.

TABLE 1

PIN	PIN Function
I/O [7:0]	Data in/out
CLE	Command latch enable
ALE	Address latch enable
CE_	Chip enable
RE_	Read enable
WE_	Write enable
WP_	Write protect
R/B_	Read/busy output

[0070] As shown in TABLE 1 above, various pin functions can correspond to designated pins in the interface. In some implementations, the NAND flash I/O interface described above can be used as the memory interface 128.

[0071] Although the flash memory devices 106a-106d are described herein as NAND-type memory devices, the flash memory devices 106a-106d also can be of the form of any one of a NOR Flash EEPROM, AND Flash EEPROM, DiNOR Flash EEPROM, Serial Flash EEPROM, DRAM, SRAM, ROM, EPROM, FRAM, MRAM, or PCRAM.

[0072] FIG. 2A shows an example of a timing diagram associated with each pin in a NAND flash interface for a READ operation. Referring to FIG. 2A, region 202 can represent a period during which one or more commands are being sent to a NAND flash memory

SAMSUNG EX1002, Page 60 of 1031

device, while region 204 can represent a period during which data are being transferred to/from the NAND flash memory device. As shown, the write enable signal "WE_" can be pulsed (e.g., at a 25-ns period) to allow row address (e.g., RA 1, RA 2, and RA 3) and column address (e.g., CA 1 and CA 2) information to be latched in the NAND flash memory device. Command "0Oh" appearing on the data in/out I/O [7:0] pin can indicate a READ address input, while command "30h" can indicate a READ start.

[0073] Other commands also can be used. For example, other memory commands can include, without limitation, READ operations, PROGRAM operations, ERASE operations, READ status operations, READ ID operations, write configuration register operations, write address operations, and reset operations. As an example, command "05h" can represent a random data READ command; command "1Oh" can represent a page program command; command "2Oh" can represent a chip erase command; command "2Ih" can represent a sector erase command; command "30h" can represent a read start command; command "35h" can represent a page read for copy command; command "39h" can represent a write device address command; command "6Oh" can represent a block erase command; command "7Oh" can represent a read status command; command "80h" can represent a serial data input (writer to buffer) command; command "85h" can represent a random data input command; command "8Fh" can represent a target address input for copy command; command "9Oh" can represent a read device type command; command "AOh" can represent a write configuration register command; command "COh" can represent a program/erase suspend command; and command "DOh" can represent a program/erase command; and command "FFh" can represent a reset command, to name a few examples.

[0074] With read enable signal "RE_" pulsing, data such as D_{0U},N, D_{OU},N+1, D_{0U},N+2, . . . D_{0Ut}M can be read from the NAND flash memory device. The read/busy output signal "R/B_" in a particular logic state can indicate whether the output is busy. For example, the read/busy output signal "R/B_" in a low logic state can indicate a busy state at the output. In this example, the read/busy output signal "R/B_" can go logic HIGH (i.e., become logic HIGH which indicates a ready state) some period of time after the last rising edge of write enable signal "WE_".

[0075] As shown in FIG. 2A, the "tR" period, which indicates a data transfer time for reading data from a cell to an internal page buffer, can span from the read start command "30h" to the rising edge of the ready/busy output signal "R/B_", and indicates a reading time for reading data. In some implementations, the "tR" period can determine the performance of a solid state drive system (e.g., solid state drive system 100) for a particular READ operation.

SAMSUNG EX1002, Page 61 of 1031

In some implementations, a long "tR" period can impose a longer wait time for a solid state controller (e.g., solid state controller 108) before the READ data can be available to the controller.

[0076] Before the data becomes ready, the NAND flash I/O interface can be idle and consume unnecessary bandwidth. Thus, it is desirable to constantly maintain the NAND flash I/O interface in a busy state to achieve maximum bandwidth. For example, if the NAND flash I/O interface is running at tRC = 25ns, the upper limit of an achievable bandwidth can be 40MB/s (e.g., based on an assumption that a 8-bit data bus is used, and that the NAND flash memory has unlimited bandwidth such that data can continuously be sent from the NAND flash memory to the solid state controller). However, this goal can be difficult to achieve given the "tR" period to read a page from a memory cell in a NAND flash memory to an internal buffer before returning the read data to a solid state controller (e.g., solid state controller 108).

[0077] Thus, in some implementations, multiple devices (e.g., multiple chip enable signals) can be bundled into a single channel while sharing a same NAND flash I/O interface such that the "tR" time can be covered as much as possible while allowing data to be available from at least one device of the same channel.

[0078] In some implementations, the data transfer time can be in the range of about 25 μs for a SLC device or in the range of about 60 μs for a MLC device. During the "tR" period, the ready/busy output signal "R/B_" can be asserted as a logical "0", indicating that the flash memory device is in a busy state, during which data, for example, cannot be written or erased.

[0079] FIG. 2B shows an example of a timing diagram associated with each pin in a NAND flash interface for a PROGRAM operation. Referring to FIG. 2B, region 206 can represent a period during which one or more commands can be sent to a flash memory device, region 208 can represent a period during which PROGRAM data can be sent to the flash memory device, and region 210 can represent a period during which a status check can be sent to the flash memory device for checking status of the flash memory device.

[0080] As shown, command "80h" appearing on the data in/out I/O [7:0] pin can indicate serial data input (e.g., $D_{In}N$ . . . $D_{In}M$). Command "1Oh" can indicate an auto program, followed by a status read as indicated by command "7Oh". I/O[0] = "0" can indicate an no-error condition, while I/O[0] = "1" can indicate that an error in auto programming has occurred.

SAMSUNG EX1002, Page 62 of 1031

[0081] Also, as already discussed, the ready/busy output signal "R/B_" can be logic low, indicating a busy state. In some implementations, the duration of the low logic state of the ready/busy output signal "R/B_" can range in the order of hundreds of μs. Also, a rising edge of the read enable signal "RE_" can trail a rising edge of the write enable signal "WE_" by a period of time. In some implementations, this period of time can be in the range of about 60 ns.

[0082] Also, as shown in FIG. 2B, the timing period "tPRG", which spans from the trailing edge to the rising edge of the ready/busy output signal "R/B_", can indicate a program time for programming data. In some implementations, the program time "tPRG" can be in the range of about 200 μs to about 700 μs for a SLC device or in the range of about 800 μs to about 3 ms for a MLC device.

[0083] In some implementations, the program time "tPRG" can be similar to the "tR" period but for programming data as opposed to reading data. During the "tPRG" period, the ready/busy output signal "R/B_" can be asserted as a logical "0", indicating that the flash memory is in a busy state, during which data, for example, cannot be read or erased. A long program time "tPRG" can reduce the performance of the flash memory device because no access to the flash memory device can typically be granted during this period.

[0084] FIG. 2C shows an example of a timing diagram associated with each pin in a NAND flash interface for a block ERASE operation. Referring to FIG. 2C, region 212 can represent a period during which one or more commands can be sent to a flash memory device, and region 214 can represent a period during which a status check can be sent to the flash memory device for checking status of the flash memory device 106a-106d.

[0085] As shown, command "6Oh" appearing on the data in/out I/O [7:0] pin can indicate a block ERASE operation, with sequential row addresses (e.g., RA 1 , RA 2 , and RA 3) being supplied. Command "DOh" can indicate a cycle 2 block ERASE operation. The block ERASE operation can be checked by a status read (command "7Oh"), where I/O[0] = "0" can indicate an no-error condition, while I/O[0] = "1" can indicate that an error in block erase has occurred.

[0086] In this example, the ready/busy output signal "R/B_" can be logic low for a period of time such as in the range of about a millisecond (e.g., with a predetermined maximum). Similarly, a rising edge of the read enable signal "RE_" can trail a rising edge of the write enable signal "WE_". As another example, a rising edge of the write enable signal "WE_" corresponding to the "DOh" command and/or a falling edge of the ready/busy output signal "R/B_" can be in the range of about 100 ns.

SAMSUNG EX1002, Page 63 of 1031

[0087] Also, as shown in FIG. 2C, the timing period "tBERS", which spans from the trailing edge to the rising edge of the ready/busy output signal "R/B_", can indicate a block erase time for erasing block data.

[0088] In some cases, a block ERASE operation need not access the NAND flash I/O interface other than sending one or more block ERASE commands (or reading status commands if not using read a ready/busy output signal "R/B_"). The performance of the solid state drive system 100 can rely on the block ERASE operation indirectly. In certain cases, at the very beginning of a block ERASE operation, the solid state drive system 100 can determine that a number of erased pages are available for data programming. As a result, the solid state drive system 100 can maintain a PROGRAM operation without the need of performing a block ERASE operation. However, as the data stored in the solid state drive system 100 is modified along with the operation of the solid state drive system 100, less pages become available for data programming such that the solid state drive system 100 would need to perform one or more block ERASE operations to vacant empty page(s) for data programming. Thus, a long "tBERS" period would yield a longer wait period for the solid state drive system 100 to honor the PROGRAM commands. In other words, the impact of a block ERASE operation on the overall drive performance depends heavily on the garbage collection mechanism handled by the firmware (e.g., firmware 124).

[0089] In some implementations, the block erase time "tBERS" can be in the range of about 1.5 ms to about 2 ms for a SLC device or in the range of about 1.5 ms to about 10 ms for a MLC device. During the "tBERS" period, the ready/busy output signal "R/B_" can be asserted as a logical "0", indicating that the flash memory is in a busy state, during which data, for example, cannot be read or written.

[0090] FIG. 3A shows an example of a horizontal data transfer scheme for a two-plane flash memory device. As shown in FIG. 3A, a flash memory device unit 300 includes four two-plane flash memory devices 300a, 300b, 300c and 300d. The two-plane flash memory devices 300a, 300b, 300c and 300d can correspond to the flash memory devices 106a-106d shown in FIG. IA, respectively.

[0091] Each of the two-plane flash memory devices 300a-300d includes plane "0" and plane "1", and receives data over four different channels; namely, "CH 0", "CH 1", "CH 2", and "CH 3". Page buffers 302, 304, 306 and 308 can be used as temporary storages for storing data received over respective channels "CH 0", "CH 1", "CH 2", and "CH 3" while PROGRAM operations initiated at a previous period are in progress.

SAMSUNG EX1002, Page 64 of 1031

[0092] Each page buffer 302-308 can be configured with a page size equivalent to a single page corresponding to a single plane (e.g., each page buffer 302-308 can have a page buffer size equal to the memory cell size. Page buffers 302, 304, 306 and 308 can be components of a solid state controller (e.g., solid state controller 108) that controls the PROGRAM/READ/ERASE operations associated with programming, reading or erasing data in plane "0" and plane "1", and communicate with the two-plane flash memory devices 300a-300d through a memory interface (e.g., memory interface 128).

[0093] During a PROGRAM operation, data words received from channels "CH 0", "CH 1", "CH 2", and "CH 3" are loaded into the page buffers 302-308. Each data word to be written can be received serially (e.g., in bit chunks), and the page buffers 302-308 can be replenished with data in a sequence of steps.

[0094] In the two-plane memory array configuration shown in FIG. 3A, one page of data can be "horizontally" programmed by loading data in the page buffers 302-308, and programming the loaded data into a corresponding page in plane "0" and plane "1". For example, assuming that a total of 8kB of data is received from the host 102 with 4kB of data directed through channel "CH 0" and 4kB of data directed through channel "CH 1", then half of the data (e.g., 4kB) can be stored in page buffer 302 (e.g., 2kB of data in plane "0" and 2kB of data in plane "1") associated with channel "CH 0" and the other half (e.g., 4kB) can be stored in page buffer 304 associated with channel "CH 1" (e.g., 2kB of data in plane "0" and 2kB of data in plane "1").

[0095] As shown in FIG. 3A, the horizontal data transfer scheme (indicated by dashed arrows) begins with loading a first set of data (e.g., up to 4kB) received from channel "CH 0" in page buffer 302, and writing the loaded data into page 310 of plane "0". Concurrently or subsequently to writing the first set of data, a second set of data from channel "CH 0" also can be loaded into page buffer 302. After completing the PROGRAM operation of the first set of data, the second set of data can be latched from page buffer 302 and programmed into page 312 of plane "1". At this time, page buffer 304 is available to receive data from channel "CH 1". Upon completing the PROGRAM operation of the second set of data from channel "CH 0", data from channel "CH 1" can be retrieved from page buffer 304 and programmed into page 314 of plane "0", after which the remaining data can be loaded from page buffer 304 and programmed into page 316 of plane "1". Similarly, upon completing the PROGRAM operation of the second set of data from channel "CH 1", data from channel "CH 2" can be retrieved from page buffer 306 and programmed into page 318 of plane "0", after which the remaining data can be loaded from the page buffer 306 and programmed into page

SAMSUNG EX1002, Page 65 of 1031

320 of plane "1". The foregoing PROGRAM operation repeats until data received from channel "CH 3" is successfully loaded and programmed into page 322 of plane "0" and page 324 of plane "1".

[0096] FIG. 3B shows an example of a timing diagram associated with the horizontal data transfer scheme shown in FIG. 3A. As shown in FIG. 3B, data for plane "0" and plane "1" of channel "CH 1" can be programmed only after the respective PROGRAM operation of data for plane "0" and plane "1" of channel "CH 0" is complete. Similarly, data for plane "0" and plane "1" of channel "CH 2" can be programmed only after the respective PROGRAM operation of data for plane "0" and plane "1" of channel "CH 1" is complete; and data for plane "0" of channel "CH 3" can be programmed only after the respective PROGRAM operation of data for plane "0" and plane "1" of channel "CH 2" is complete. Because of these timing constraints imposed by the horizontal data transfer scheme, PROGRAM operations cannot start at the same time in parallel across channels "CH 0", "CH 1", "CH 2", and "CH 3", which can lengthen the programming time "tPRG". A long programming time, however, can prolong the period during which a flash memory device cannot be accessed, even if the firmware (e.g., firmware 124) has pending access request to access the flash memory device.

[0097] Further, depending on the load of data transmitted over channels "CH 0", "CH 1", "CH 2", and "CH 3" as well as the size of the pages 310-324, the need to increase the size of the page buffers 302-308 can arise in order to accommodate the stringent channel buffer size requirement. For example, where each page is 4kB, the size of each page buffer 302-308 would require a minimum of 8kB for each of the two-plane flash memory devices 300a-300d (e.g., 4kB for each page/plane) with a total of 32kB of flash page buffer size. As another example, for a eight-plane flash memory device with each plane corresponding to a single page of 4kB, a total of 128kB flash page buffer size would be required (e.g., 32kB to accommodate each channel for four channels).

[0098] If the page buffers 302-308 cannot accommodate this flash page buffer size, then the size of the page buffers 302-308 would need to be increased in order to fully and efficiently execute the PROGRAM operations according to the horizontal data transfer order shown in FIG. 3A. However, in some cases, even when the page buffer size is increased, the performance of the flash memory device is not necessary improved. Using the two-plane flash memory device 300a-300d shown in FIG. 3A as an example, increasing the size of the page buffers 302-308 (e.g., increasing each page buffer 302-308 from one-page buffer size or 4kB to two-page buffer size or 8kB) does not necessarily improve the overall PROGRAM

SAMSUNG EX1002, Page 66 of 1031

speed of the two-plane flash memory device 300a-300d because of the limitations imposed by the horizontal data transfer scheme. Since data from channel "CH 1" cannot be programmed until after the PROGRAM operation for data from channel "CH 0" is completed, the timing period "tPRG" to complete the PROGRAM operations for channels "CH 0", "CH 1", "CH 2", and "CH 3" remains long. A long "tPRG" period can impose a longer wait time for a solid state controller (e.g., solid state controller 108) before the PROGRAM data can be accessed by the solid state controller, and undesirably reduces the access speed at which the flash memory devices can be accessed.

[0099] To allow pages 310-324 to be programmed without a need to increase the size of page buffers 302-308, in some implementations, a vertical data transfer scheme can be used to program the pages 310-324 of plane "0" and plane "1". In some implementations, the vertical data transfer scheme allows pages of a first plane to be programmed prior to programming pages of other planes. After programming the pages of the first plane, programming of the pages of a subsequent plane can commence. FIG. 4A shows an example of a vertical data transfer scheme.

[00100] Referring to FIG. 4A, a two-plane flash memory device 400 can include plane "0" and plane "1". Although only two planes are shown, one skilled in the art would recognize that the two-plane flash memory device 400 can include more than two planes, and the vertical data transfer scheme as will be discussed below is equally applicable.

[00101] Similar to the two-plane flash memory device 300a-300d shown in FIG. 3, the two-plane flash memory device 400 can receive data over channels "CH 0", "CH 1", "CH 2", and "CH 3". Page buffers 402, 404, 406 and 408 can be used as temporary storages for storing data received over respective channels "CH 0", "CH 1", "CH 2", and "CH 3".

[00102] Data can be loaded sequentially into each of the page buffers 402-408. Specifically, page buffer 402 can receive a first group of data from channel "CH 0" to be programmed into page 410a of plane "0", and a second group of data from channel "CH 0" to be programmed into page 410b of plane "1"; page buffer 404 can receive a first group of data from channel "CH 1" to be programmed into page 412a of plane "0", and a second group of data from channel "CH 1" to be programmed into page 412b of plane "1"; page buffer 406 can receive a first group of data from channel "CH 2" to be programmed into page 414a of plane "0", and a second group of data from channel "CH 2" to be programmed into page 414b of plane "1"; and page buffer 408 can receive a first group of data from channel "CH 3" to be programmed into page 416a of plane "0", and a second group of data from channel "CH 3" to be programmed into page 416b of plane "1".

SAMSUNG EX1002, Page 67 of 1031

[00103] As discussed above, data stored in the page buffers 402-408 can be programmed into respective pages of plane "0" and plane "1" using a vertical data transfer scheme. As indicated by the dashed lines, after the first group of data from channel "CH 0" is loaded into the page buffer 402, the first group of data from channel "CH 0" can be programmed into page 410a of plane "0". During this programming phase, data stored in the page buffers 404-408 also can be loaded and programmed into the respective pages 412a, 414a and 416a of plane "0". As an example, after loading the first group of data from channel "CH 1" into page buffer 404, the first group of data from channel "CH 1" can immediately be programmed into page 412a of plane "0" in parallel with the PROGRAM operation associated with channel "CH 0". Similarly, the first group of data from channel "CH 2" and channel "CH 3" can immediately be loaded and programmed into the respective pages 414a and 416a of plane "0".

[00104] After programming the first group of data from channels "CH 0", "CH 1", "CH 2", and "CH 3" into respective pages 410a, 412a, 414a and 416a of plane "0", the second group of data from channels "CH 0", "CH 1", "CH 2", and "CH 3" can be loaded into the page buffers 402-408 and programmed into respective pages 410b, 412b, 414b and 416b of plane "1". For example, after programming the first group of data from channel "CH 3" into page 416a of plane "0", the second group of data from channel "CH 0" can be loaded and programmed into page 410b of plane "1". At this time, the second group of data from channels "CH 1", "CH 2" and "CH 3" also are loaded into the page buffers 404-408 respectively. After loading the second group of data from the page buffers 404-408, the second group of data are programmed into pages 412b, 414b and 416b of plane "1". The vertical data transfer scheme allows the first group of data to be programmed into the pages of plane "0" prior to commencing the PROGRAM operations of the second group of data into the pages of plane "1".

[00105] In some implementations, pages 410a, 412a, 414a and 416a of plane "0" can be programmed in parallel, and pages 410b, 412b, 414b and 416b of plane "1" can be programmed in parallel. One skilled in the art would recognize that programming the foregoing pages does not mean that the respective PROGRAM operations start at exactly the same time. For example, the PROGRAM operations associated with programming pages 410a, 412a, 414a and 416a of plane "0" each can be initialized after a predetermined timing offset. As an example, the PROGRAM operation associated with programming page 410a can be initialized at a particular reference time. After a predetermined timing period has elapsed (e.g., from the reference time), the PROGRAM operation associated with

SAMSUNG EX1002, Page 68 of 1031

programming page 412a can commence. In some implementations, this predetermined timing offset can depend on the time associated with loading the data into a respective page buffer. For example, the timing difference between initializing the PROGRAM operation associated with programming page 410a and the PROGRAM operation associated with programming page 412a is the time that the solid state drive would need to initialize loading of the data from page buffer 404 after the loading of data from page buffer 402 has begun. Similarly, the PROGRAM operations associated with programming pages 410b, 412b, 414b and 416b of plane "1" each can be initialized after a predetermined timing offset.

[00106] FIG. 4B shows an example of a timing diagram associated with the vertical data transfer scheme shown in FIG. 4A. As shown in FIG. 4B, PROGRAM operations for data received from channels "CH 0", "CH 1", "CH 2", and "CH 3" can be executed in parallel. Specifically, data from channels "CH 0", "CH 1", "CH 2", and "CH 3" can be programmed into plane "0" in parallel. After pages of plane "0" are programmed, the solid state controller 108 can begin programming the pages of plane "1" as indicated by the vertical data transfer scheme shown in FIG. 4A. Compared to the timing diagram for the horizontal data transfer scheme shown in FIG. 3B, the vertical data transfer scheme significantly reduces the PROGRAM timing period "tPRG" and improves the overall speed of the PROGRAM operations because the vertical data transfer scheme allows all channels to be programmed in parallel. The vertical data transfer scheme can reduce the PROGRAM timing period "tPRG" to less than 200 μs for a SLC device or less than 800 μs for a MLC device.

[00107] FIG. 5 shows an example of a process 500 for executing a vertical data transfer scheme. The process 500 can be performed, for example, by the solid state system 100 or the solid state controller 104, and for clarity of presentation, the description that follows uses the solid state controller 104 as the basis of examples for describing the process 500. However, another system or combination of devices and systems also can be used to perform the process 500.

[00108] Referring to FIG. 5, at 502, first data and second data can be received. At 504, the first data can be stored in a first page buffer, and the second data can be stored in a second page buffer. At 506, the first data stored in the first page buffer can be programmed into a first page associated with a first memory plane. At 508, the second data stored in the second page buffer can be programmed into a second page associated with the first memory plane while programming the first data.

SAMSUNG EX1002, Page 69 of 1031

[00109] In some implementations, operations 502-508 can be performed in the order listed or in parallel (e.g., by the same or a different process, substantially or otherwise non-serially) to achieve the same result. Operations 502-508 also can be performed by the same or different entities or systems.

[00110] Although the foregoing implementations are described in terms of programming data from channels "CH 0", "CH 1", "CH 2", and "CH 3" in parallel, PROGRAM operations also can begin after the PROGRAM operation for a preceding channel has ended. For example, the first group of data from channel "CH 1" can commence upon completing the PROGRAM operation of page 410a using data from channel "CH 0". As another example, the first group of data from channel "CH 2" can commence upon completing the PROGRAM operation of page 412a using data from channel "CH 1". As yet another example, the first group of data from channel "CH 3" can commence upon completing the PROGRAM operation of page 414a using data from channel "CH 2".

[00111] While only two planes are described in FIG. 4, the vertical data transfer scheme also can be applied to multi-plane memory devices with more than two planes. For example, the vertical data transfer scheme can be applied to a multi-plane memory device with four planes. In this example, after completing the PROGRAM operation of page 416b, the vertical data transfer scheme can proceed to program a third set of group data from channel "CH 0" into a first page of the third plane (i.e., plane "2").

[00112] The example solid state drive system 100 shown in FIG. IA is an exemplary system configuration, which is shown purely for the sake of conceptual clarity. Any other suitable memory system configuration also can be used. Elements that are not necessary for understanding the principles of the subject matter described herein, such as various interfaces, addressing circuits, timing and sequencing circuits and debugging circuits, have been omitted from the figure for clarity.

[00113] The solid state drive system 100 or the solid state drive 104 including the solid state controller 108 and the memory controller 130 can be mounted in a package. Function blocks and peripheral circuits can be further included in the package. For example, the solid state drive system 100 or the solid state drive 104 can be mounted in a package such as Package on Package (PoP), Ball Grid Arrays (BGAs), Chip Scale Packages (CSPs), Plastic Leaded Chip Carrier (PLCC), Plastic Dual In-Line Package (PDIP), Die in Waffle Package, Die in Wafer Form, Chip On Board (COB), Ceramic Dual In-Line Package (CERDIP), Plastic Metric Quad Flat Package (MQFP), Thin Quad Flat Package (TQFP), Small Outline Integrated Circuit (SOIC), Shrink Small Outline Package (SSOP), Thin Small Outline

SAMSUNG EX1002, Page 70 of 1031

Package (TSOP), Thin Quad Flat Package (TQFP), System In Package (SIP), Multi Chip Package (MCP), Wafer-level Fabricated Package (WFP), Wafer-Level Processed Stack Package (WSP), and the like.

[00114] The solid state drive system 100 or the solid state drive 104 including the solid state controller 108 and the memory controller 130 also can reside in a memory card. The memory card can communicate with an external device (e.g., a host) through at least one interface among Universal Serial Bus (USB), Multimedia Card (MMC), Peripheral Component Interconnect-Express (PCI-E), Serial Advanced Technology Attachment (SATA), Parallel Advanced Technology Attachment (PATA), Small Computer System Interface (SCSI), Enhanced Small Device Interface (ESDI), Integrated Drive Electronics (IDE), and the like.

[00115] The solid state drive system 100 or the solid state drive 104 including the solid state controller 108 and the memory controller 130 also can reside in a mobile device such as a cellular phone, a personal digital/data assistant (PDA), a digital camera, a portable game console and MP3 player. Where the solid state drive system 100 or the solid state drive 104 is implemented in a mobile device, the flash memory devices 106a-106d can be internal or external to the mobile device, and can store codes for an operation of the mobile device as well as data.

[00116] The solid state drive system 100 or the solid state drive 104 including the solid state controller 108 and the memory controller 130 can be adaptable to a home application such as a high definition television (HDTV), a digital video disk, a digital versatile disc (DVD), a router, Global Positioning System (GPS), and the like.

[00117] The solid state drive system 100 or the solid state drive 104 including the solid state controller 108 and the memory controller 130 can be adaptable to a computing system. Where the solid state drive system 100 or the solid state drive 104 is implemented in a computing system, the solid state drive system 100 or the solid state drive 104 can further include a microprocessor electrically coupled to a bus, a user interface, and a modem such as a baseband chipset. The flash memory devices 106a-106d can be provided as internal or external components to the computing system, and can store data that can be processed by the microprocessor through the memory controller 130. In a case where the computing system is a mobile device, a battery for supply of an operating voltage of the computing system can be further included. It will be understood to those skilled in the art that the computing system can further include an application chipset, a camera image processor (CIS), and/or a mobile DRAM.

SAMSUNG EX1002, Page 71 of 1031

[00118] The flash memory devices 106a-106d are non-volatile memory devices capable of maintaining stored data even though power is off. Thus, it will be understood to those skilled in the art that the flash memory devices 106a-106d can be adaptable to various devices, apparatuses, and systems as well as applications.

[00119] As described above, multi-planes of a flash memory device can be programmed using a vertical data transfer scheme to reduce the PROGRAM time "tPRG" such that the performance of the flash memory devices 106a-106d and the solid state system 100 can be improved without the need to increase the size of the page buffers 132a-132d.

[00120] A few implementations have been described in detail above, and various modifications are possible. The disclosed subject matter, including the functional operations described in this specification, can be implemented in electronic circuitry, computer hardware, firmware, software, or in combinations of them, such as the structural means disclosed in this specification and structural equivalents thereof, including potentially a program operable to cause one or more data processing apparatus to perform the operations described (such as a program encoded in a computer-readable medium, which can be a memory device, a storage device, a machine-readable storage substrate, or other physical, machine-readable medium, or a combination of one or more of them).

[00121] The term "system" encompasses all apparatus, devices, and machines for processing data, including by way of example a programmable processor, a computer, or multiple processors or computers. The system can include, in addition to hardware, code that creates an execution environment for the computer program in question, e.g., code that constitutes processor firmware, a protocol stack, a database management system, an operating system, or a combination of one or more of them.

[00122] A program (also known as a computer program, software, software application, script, or code) can be written in any form of programming language, including compiled or interpreted languages, or declarative or procedural languages, and it can be deployed in any form, including as a stand alone program or as a module, component, subroutine, or other unit suitable for use in a computing environment. A program does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that holds other programs or data (e.g., one or more scripts stored in a markup language document), in a single file dedicated to the program in question, or in multiple coordinated files (e.g., files that store one or more modules, sub programs, or portions of code). A program can be deployed to be executed on one computer or on multiple computers that are

SAMSUNG EX1002, Page 72 of 1031

located at one site or distributed across multiple sites and interconnected by a communication network.

[00123] While this specification contains many specifics, these should not be construed as limitations on the scope of what can be claimed, but rather as descriptions of features that can be specific to particular implementations. Certain features that are described in this specification in the context of separate implementations can also be implemented in combination in a single implementation. Conversely, various features that are described in the context of a single implementation can also be implemented in multiple implementations separately or in any suitable subcombination. Moreover, although features can be described above as acting in certain combinations and even initially claimed as such, one or more features from a claimed combination can in some cases be excised from the combination, and the claimed combination can be directed to a subcombination or variation of a subcombination.

[00124] Similarly, while operations are depicted in the drawings in a particular order, this should not be understood as requiring that such operations be performed in the particular order shown or in sequential order, or that all illustrated operations be performed, to achieve desirable results. In certain circumstances, multitasking and parallel processing can be advantageous. Moreover, the separation of various system components in the implementations described above should not be understood as requiring such separation in all implementations.

[00125] Other implementations fall within the scope of the following claims.

SAMSUNG EX1002, Page 73 of 1031

WHAT IS CLAIMED IS:

1. A method comprising:

receiving first data and second data;

storing the first data in a first page buffer and the second data in a second page buffer;

programming the stored first data into a first page associated with a memory plane of a multi-plane memory device; and

programming the stored second data into a second page associated with the memory plane in parallel with programming the stored first data.

2. The method of claim 1, where receiving the first data includes receiving the first data over a first channel; and

where receiving the second data includes receiving the second data over a second channel different from the first channel.

3. The method of claim 1, where receiving the first data includes receiving a first subset of the first data and a second subset of the first data; and

where receiving the second data includes receiving a first subset of the second data and a second subset of the second data.

4. The method of claim 3, where programming the stored first data into the first page associated with the memory plane includes programming the first subset of the first data into the first page associated with the memory plane; and

where programming the stored second data into the second page associated with the memory plane includes programming the first subset of the second data into the second page associated with the memory plane.

5. The method of claim 4, where the memory plane is a first memory plane, the method further comprising:

programming the second subset of the first data into a first page associated with a second memory plane of the multi-plane memory device; and

programming the second subset of the second data into the second page associated with the second memory plane.

SAMSUNG EX1002, Page 74 of 1031

6. The method of claim 5, where programming the second subset of the first data into the first page associated with the second memory plane is performed after programming the first subset of the second data into the second page associated with the first memory plane.

7. The method of claim 5, where programming the second subset of the second data into the second page associated with the second memory plane is performed after programming the second subset of the first data into the first page associated with the second memory plane.

8. The method of claim 1, where programming the stored second data into a second page associated with the memory plane in parallel with programming the stored first data includes initializing a program operation associated with the stored second data before completing the programming of the stored first data.

9. A device comprising:

a first page buffer configured to receive first data from a first channel and store the first data;

a second page buffer configured to receive second data from a second channel and store the second data; and

circuitry configured to program the stored first data into a first page associated with a memory plane and the stored second data into a second page associated with the memory plane in parallel.

10. The device of claim 9, where the first data includes a first subset of the first data and a second subset of the first data; and

where the second data includes a first subset of the second data and a second subset of the second data.

SAMSUNG EX1002, Page 75 of 1031

11. The device of claim 10, where the circuitry is configured to program the first subset of the first data into the first page associated with the memory plane, and the first subset of the second data into the second page associated with the memory plane.

12. The device of claim 11, where:

the memory plane is a first memory plane; and

the circuitry is configured to program the second subset of the first data into a first page associated with a second memory plane, and the second subset of the second data into the second page associated with the second memory plane.

13. The device of claim 12, where the circuitry is configured to program the second subset of the first data into the first page associated with the second memory plane after the first subset of the second data is programmed into the second page associated with the first memory plane.

14. The device of claim 12, where the circuitry is configured to program the second subset of the second data into the second page associated with the second memory plane after the second subset of the first data is programmed into the first page associated with the second memory plane.

15. The device of claim 9, where the circuitry is configured to initialize a program operation associated with programming the stored second data before completing a program operation associated with programming the stored first data.

16. A system comprising:

a memory array having a memory plane;

a first page buffer to receive first data from a first channel and store the first data;

a second page buffer to receive second data from a second channel and store the second data; and

a controller configured to program the stored first data into a first page associated with the memory plane and the stored second data into a second page associated with the memory plane in parallel.

SAMSUNG EX1002, Page 76 of 1031

17. The system of claim 16, where the controller is configured to initialize a program operation associated with the stored second data before completing the programming of the stored first data.

18. The system of claim 16, where:

the memory plane is a first memory plane;

the memory array includes a second memory plane;

the first data includes a first subset and a second subset; and

the controller is configured to program the first subset into the first page associated with the first memory plane and the second subset into a first page associated with the second memory plane.

19. The system of claim 18, where:

the second data includes a first subset and a second subset; and

the controller is configured to program the first subset of the second data into the second page associated with the first memory plane and the second subset of the second data into a second page associated with the second memory plane.

20. The system of claim 19, where the controller is further configured to initialize a program operation associated with the first subset of the second data before initializing a program operation associated with the second subset of the first data but after initializing a program operation associated with the first subset of the first data.

SAMSUNG EX1002, Page 77 of 1031

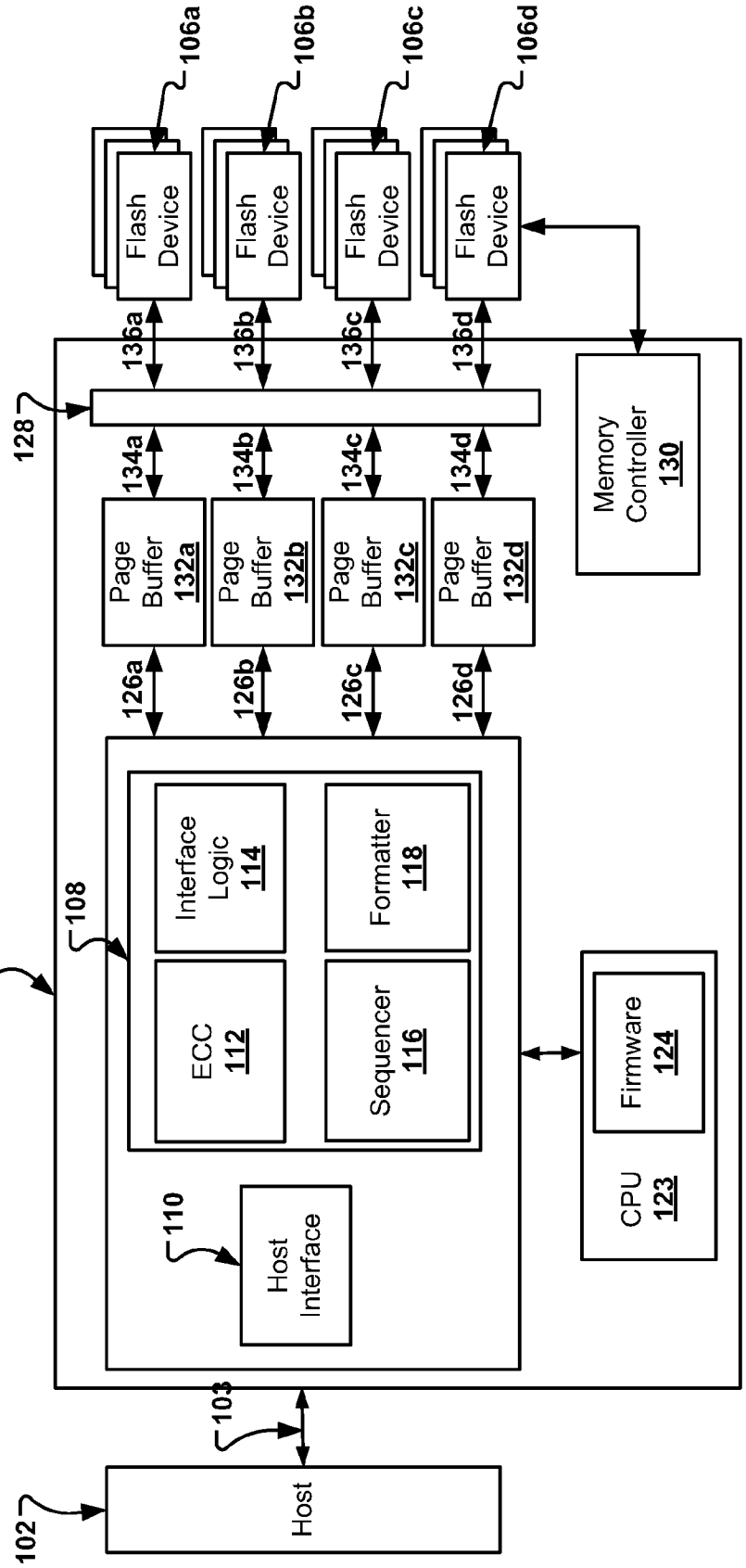

100
102
103
104
106a
106b
106c
106d
108
110
112
114
116
118
123
124
126a
126b
126c
126d
128
130
132a
132b
132c
132d
134a
134b
134c
134d
136a
136b
136c
136d
Flash Device
Flash Device
Flash Device
Flash Device
Memory Controller 130
Page Buffer 132a
Page Buffer 132b
Page Buffer 132c
Page Buffer 132d
Interface Logic 114
Formatter 118
ECC 112
Sequencer 116
Host Interface
CPU 123
Firmware 124
Host

FIG. 1A

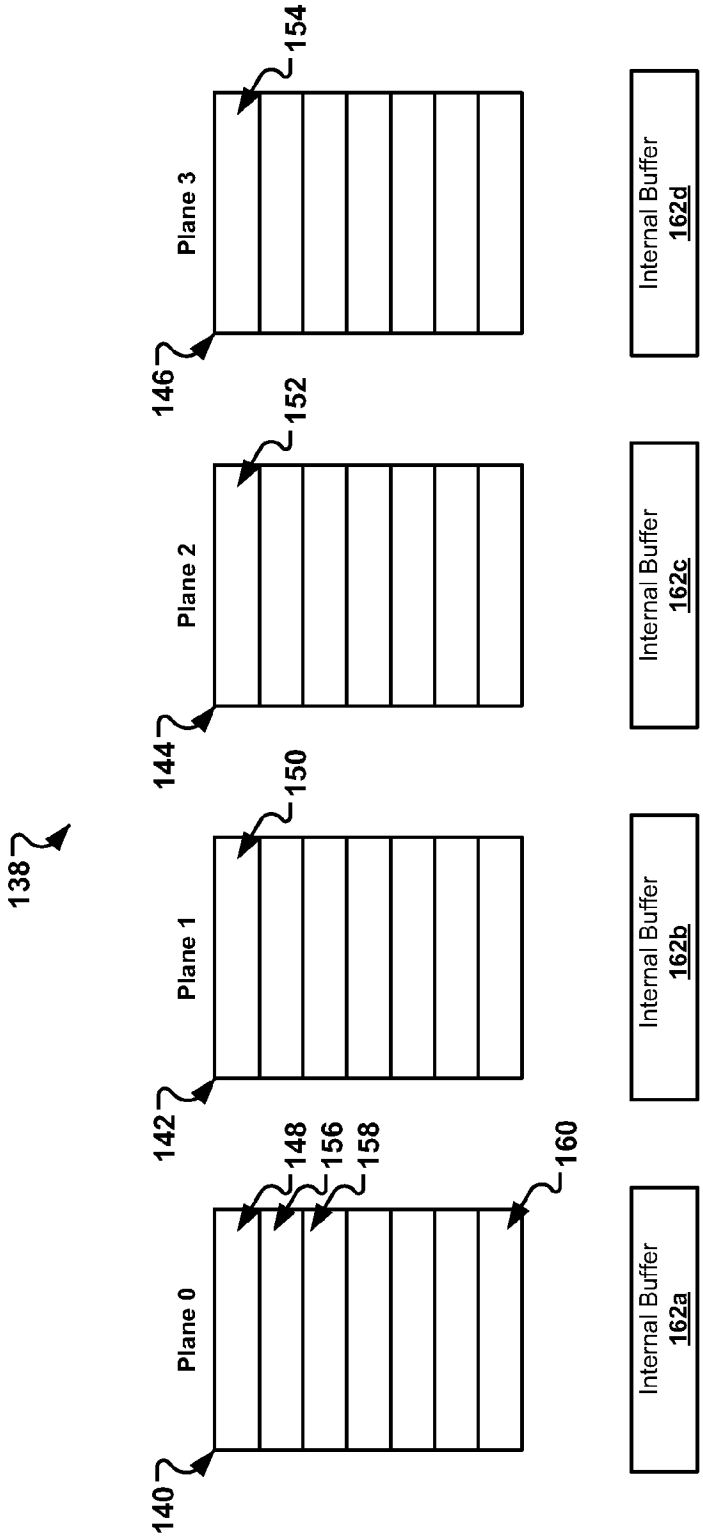

138
Plane 0
140
148
156
158
160
Plane 1
142
150
Plane 2
144
152
Plane 3
146
154
Internal Buffer 162a
Internal Buffer 162b
Internal Buffer 162c
Internal Buffer 162d

FIG. 1B

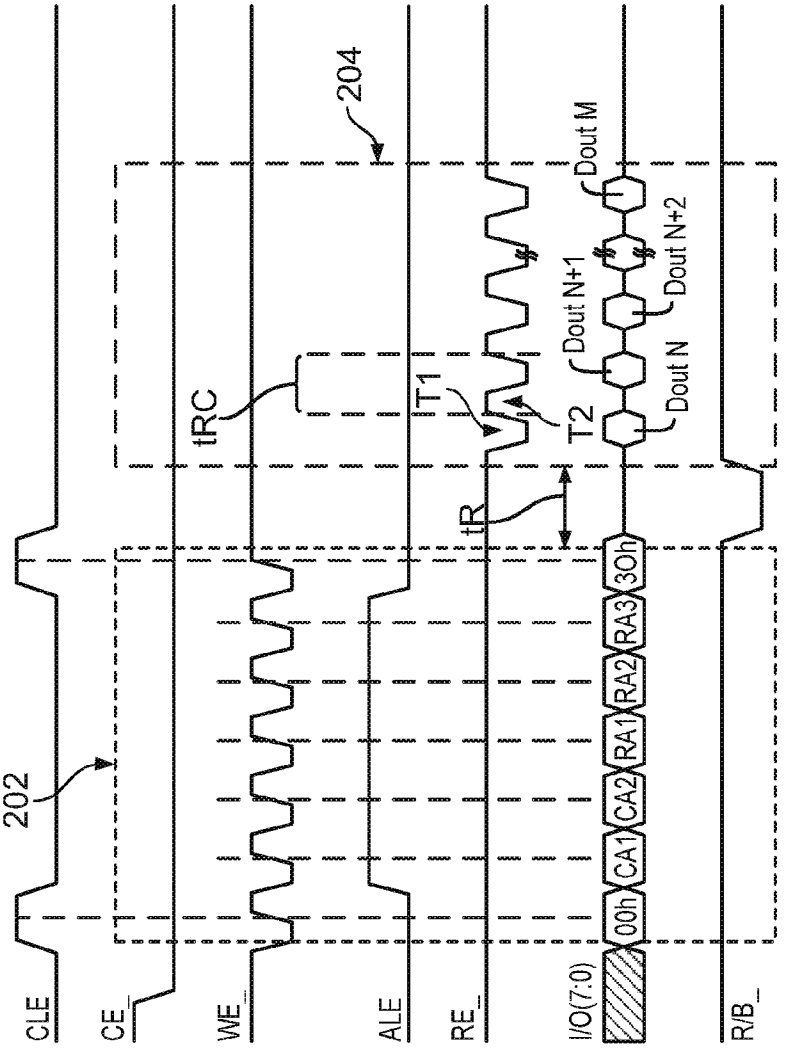

202
204
CLE
CE_
WE_
ALE
RE_
I/O(7:0)
R/B_
tRC
T1
T2
tR
00h
CA1
CA2
RA1
RA2
RA3
3Oh
Dout N
Dout N+1
Dout N+2
Dout M

FIG. 2A

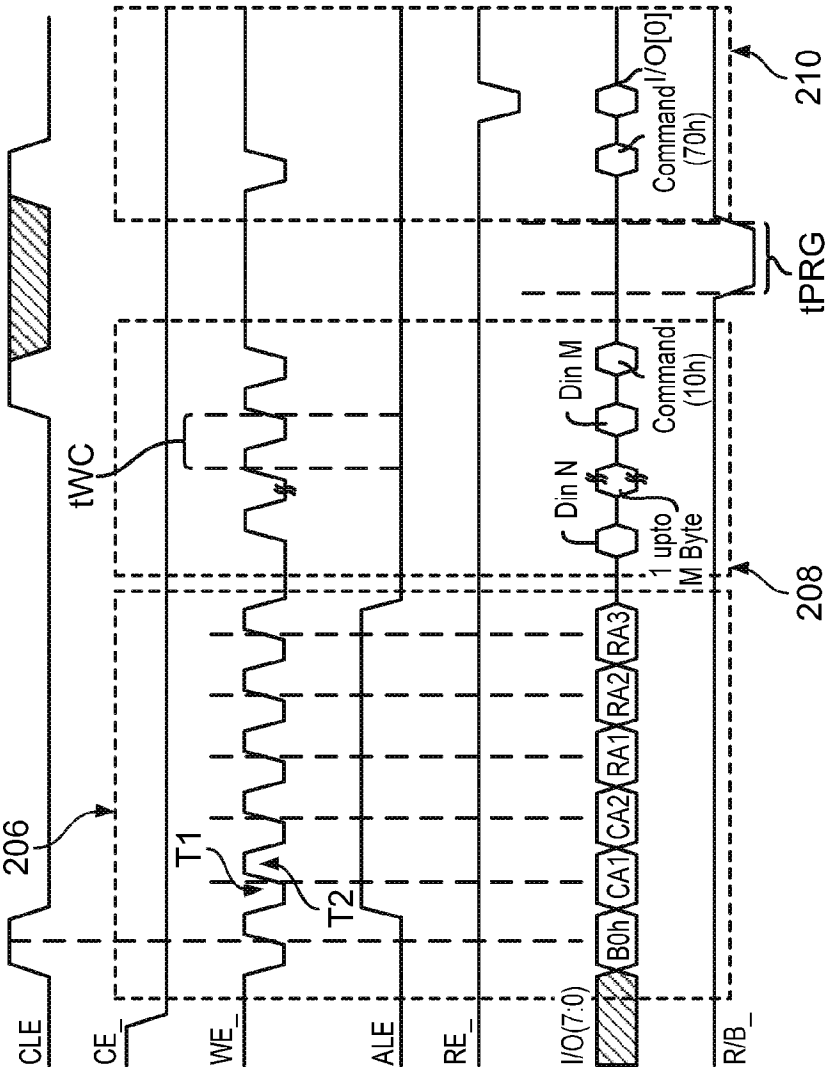

CLE
CE_
WE_
ALE
RE_
I/O(7:0)
R/B_
206
T1
T2
tWC
B0h
CA1
CA2
RA1
RA2
RA3
Din N
1 upto M Byte
Din M
Command (10h)
tPRG
Command (70h)
I/O[0]
208
210

FIG. 2B

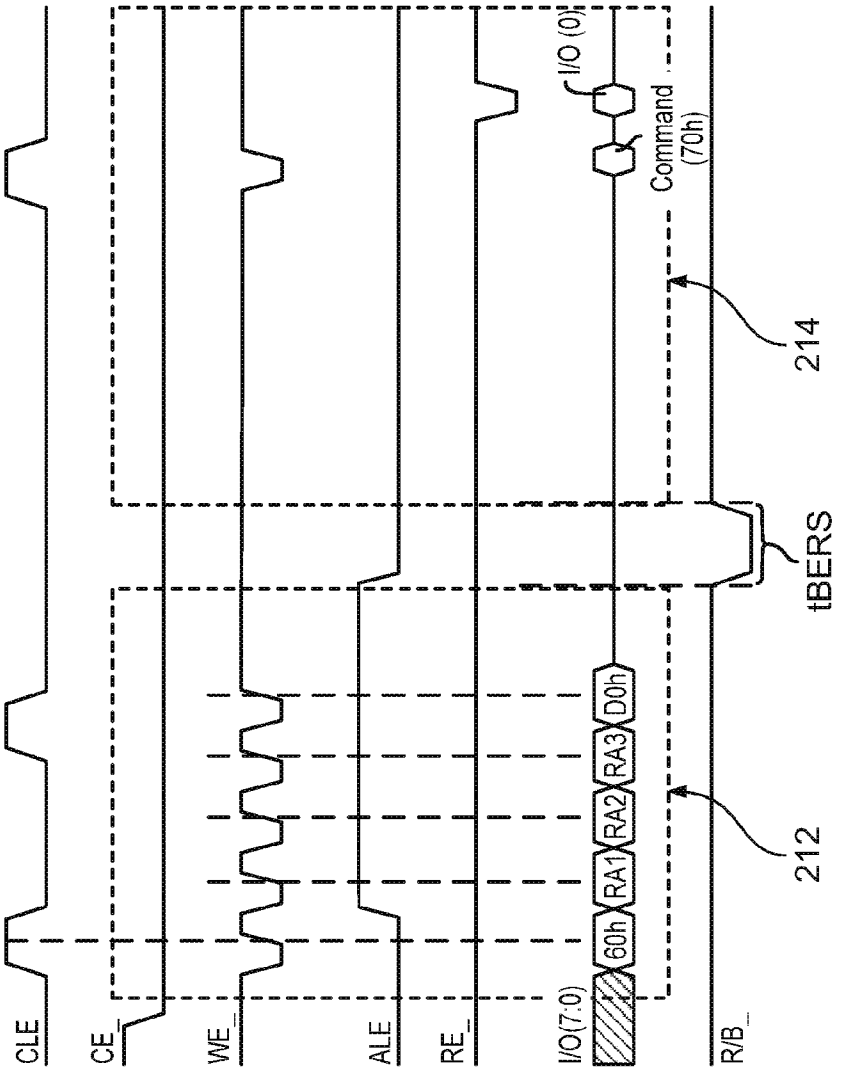

CLE
CE_
WE_
ALE
RE_
I/O(7:0)
60h
RA1
RA2
RA3
D0h
I/O (0)
Command
(70h)
214
212
tBERS
R/B_

FIG. 2C

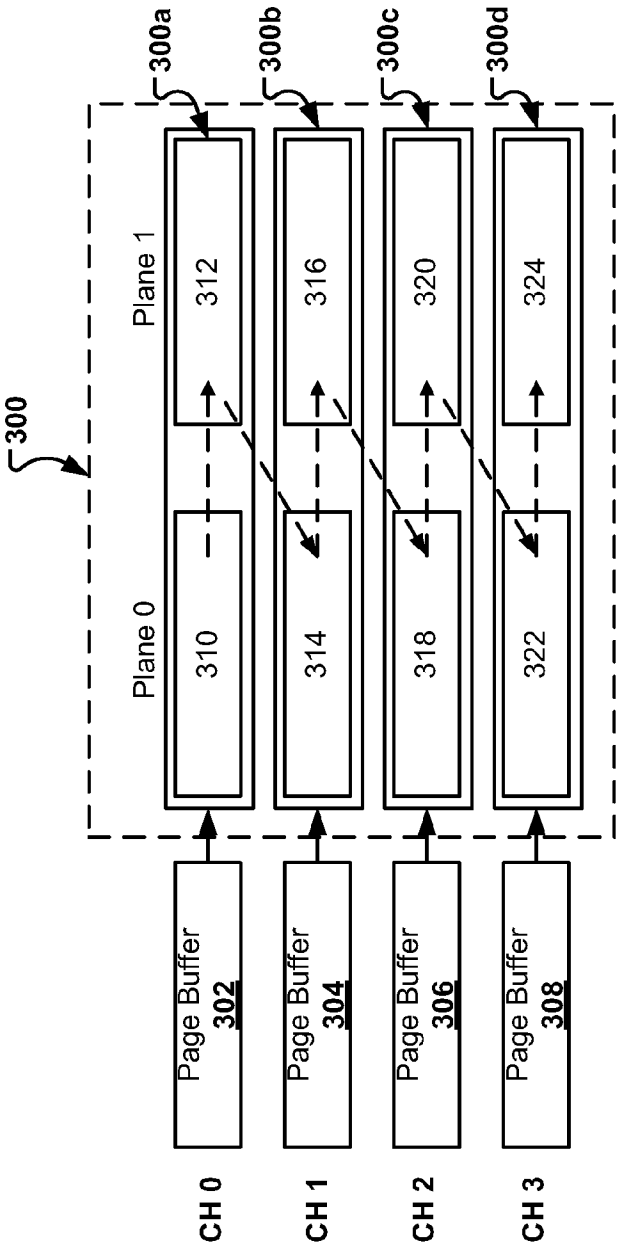

300
300a
300b
300c
300d
Plane 1
Plane 0
312
316
320
324
310
314
318
322
Page Buffer 302
Page Buffer 304
Page Buffer 306
Page Buffer 308
CH 0
CH 1
CH 2
CH 3

FIG. 3A

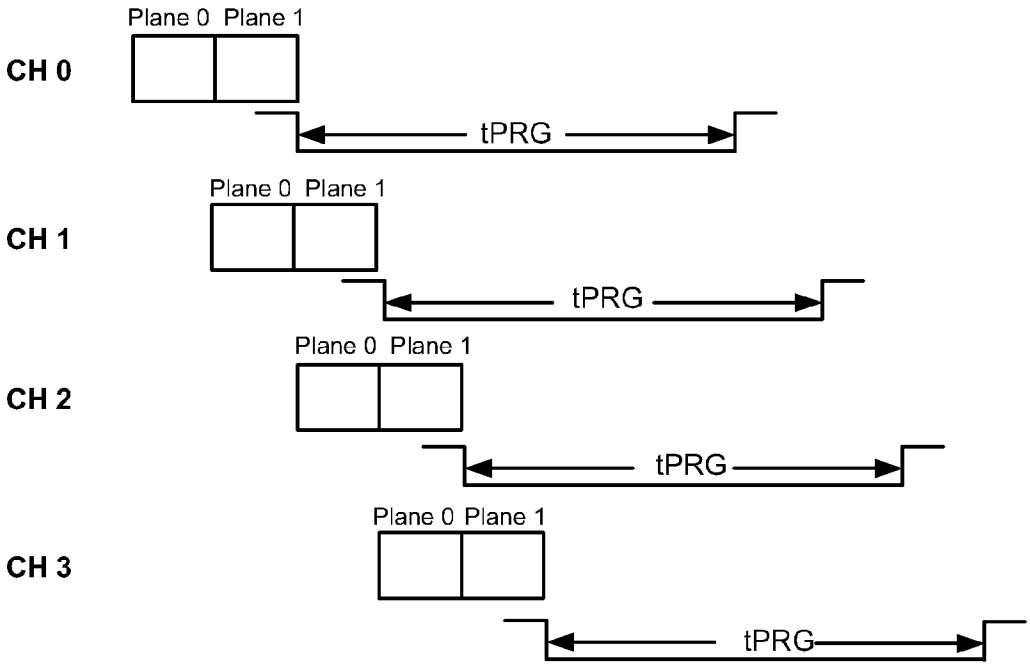

CH 0
Plane 0 Plane 1
tPRG
CH 1
Plane 0 Plane 1
tPRG
CH 2
Plane 0 Plane 1
tPRG
CH 3
Plane 0 Plane 1
tPRG

FIG. 3B

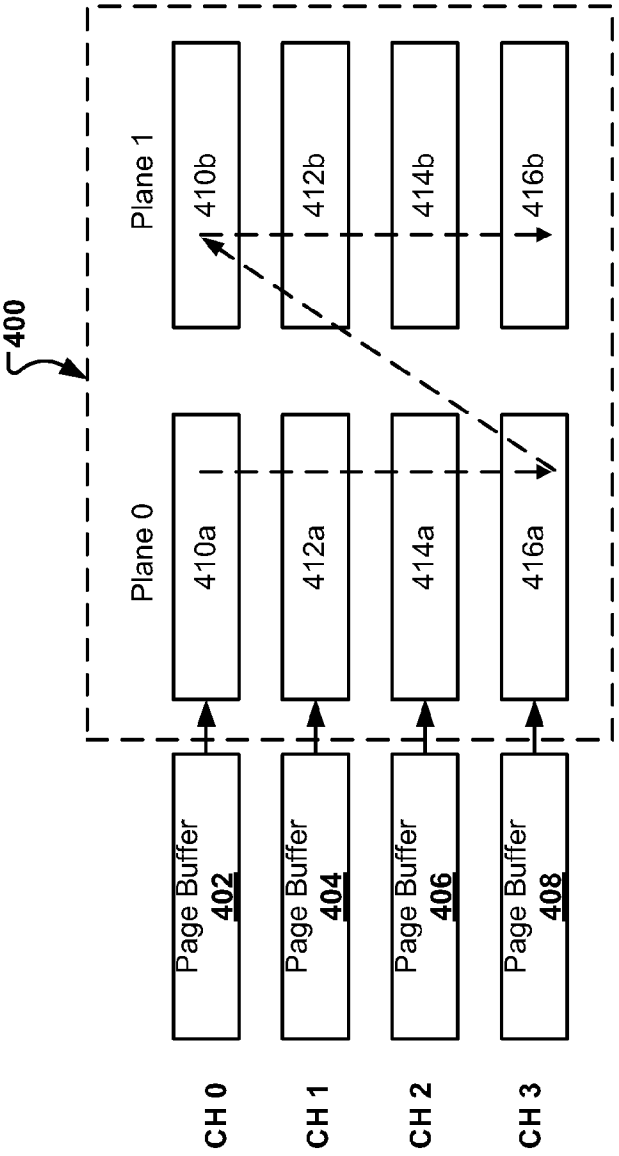

400
Plane 1
Plane 0
410b
412b
414b
416b
410a
412a
414a
416a
Page Buffer 402
Page Buffer 404
Page Buffer 406
Page Buffer 408
CH 0
CH 1
CH 2
CH 3

FIG. 4A

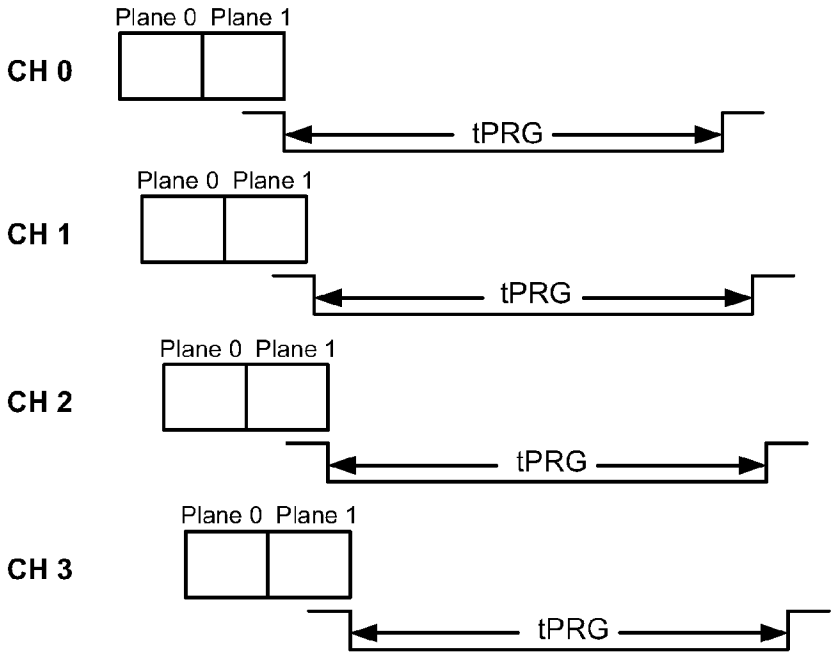

Plane 0 Plane 1
CH 0
tPRG
Plane 0 Plane 1
CH 1
tPRG
Plane 0 Plane 1
CH 2
tPRG
Plane 0 Plane 1
CH 3
tPRG

FIG. 4B

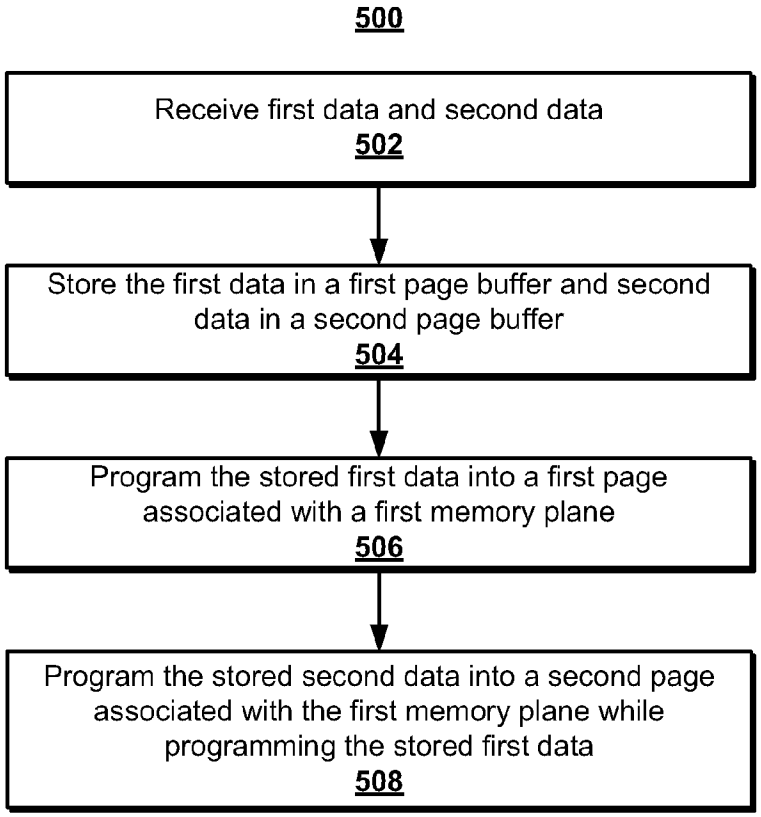

500
Receive first data and second data
502
Store the first data in a first page buffer and second data in a second page buffer
504
Program the stored first data into a first page associated with a first memory plane
506
Program the stored second data into a second page associated with the first memory plane while programming the stored first data
508

FIG. 5

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2009/055632

A. CLASSIFICATION OF SUBJECT MATTER
INV. G11C7/10 G11C16/10 G06F13/16

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)
G11C G06F

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

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

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X Y	US 2008/101120 A1 (PARK KI-TAE [KR] ET AL) 1 May 2008 (2008-05-01) paragraph [0059] - paragraph [0093]; figures 4-11	1,3-8 2,9-20
X	US 2008/049513 A1 (JEONG SEONG-HUN [KR] ET AL) 28 February 2008 (2008-02-28) paragraph [0039] - paragraph [0083]; figures 2,4-8	1
Y A	US 2003/191912 A1 (DORTU JEAN-MARC [DE]) 9 October 2003 (2003-10-09) paragraph [0031] - paragraph [0032]; figures 1,2 paragraph [0039]; figure 5	2,9-20 1,3-8

-/--

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents :

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

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

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search	Dats of mailing of the international search report
18 December 2009	04/01/2010

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	**Cummings , Anthony**

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2009/055632

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No
A	US 2007/061538 A1 (CHANG SEUNG H [KR] ET AL) 15 March 2007 (2007-03-15) paragraph [0026] - paragraph [0038]; figures 4-6	1-20
A	US 5 519 847 A (FANDRICH MICKEY L [US] ET AL) 21 May 1996 (1996-05-21) figures 1 8	1-8

SAMSUNG EX1002, Page 89 of 1031

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2009/055632

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2008101120 A1	01-05-2008	JP 2008108418 A	08-05-2008
		KR 100806119 B1	22-02-2008
US 2008049513 A1	28-02-2008	JP 2008052895 A	06-03-2008
us 2003191912 A1	09-10-2003	DE 10215362 A1	30-10-2003
us 2007061538 A1	15-03-2007	CN 1933025 A	21-03-2007
		JP 2007080475 A	29-03-2007
us 5519847 A	21-05-1996	JP 3737529 B2	18-01-2006
		JP 7141247 A	02-06-1995

1. A New Technique for the Development and Maintenance of Parametric Programs

IPCOM000148986DSoftware Patent Institute2007-04-123The IP.com Prior Art Database (backfile addition)

2. Mapping structures for flash memories: techniques and open problems

IEEE XploreIEEE Conferences2005-01-013IEEE International Conference on Software - Science, Technology & Engineering (SwSTE'05) (Page(s): 83-92)

3. IEEE Standard for Information Technology--POSIX(TM) Ada Language Interfaces--Part 1: Binding for System Application Program Interface (API)--Amendment 1: Realtime Extensions

IEEE XploreIEEE Standards1997-11-303IEEE Std 1003.5b-1996 (Includes IEEE Std 1003.5-1992) (Page(s): 1-550)

4. Tape Drive Attachment Support

IPCOM000052756DIBM TDB Archive1981-07-013IBM Technical Disclosure Bulletin (TDB 07-81 p971-975)

5. A Concurrent General Purpose Operator Interface

IEEE XploreIEEE Periodicals1984-11-013IEEE Transactions on Software Engineering

6. IEEE Standard for Information Technology - POSIX(R) Ada Language Interfaces - Part 1: Binding for System Application Program Interface (API) - Amendment 2: Protocol-Independent Interfaces

IEEE XploreIEEE Standards1999-01-013IEEE Std 1003.5, 1999 Edition (Page(s): 0_1)

7. Ozone (O3): An Out-of-Order Flash Memory Controller Architecture

IEEE XploreIEEE Periodicals2010-10-073IEEE Transactions on Computers

8. Interactive Graphics in Data Processing: A language for three-dimensional geometry

IEEE XploreIBM Periodicals1968-01-013IBM Systems Journal

9. An Application of a Large-Image Memory Graphics Workstation

IEEE XploreIEEE Periodicals1985-02-013IEEE Computer Graphics and Applications

10. Conference Record of the Twenty First IEEE Photovoltaic Specialists Conference - 1990 (Cat. No.90CH2838-1)

IEEE XploreIEEE Conferences1990-01-013IEEE Conference on Photovoltaic Specialists (Page(s): 0_1)

11. Unit Testing of Flash Memory Device Driver through a SAT-Based Model Checker

IEEE XploreIEEE Conferences2008-09-0152008 23rd IEEE/ACM International Conference on Automated Software Engineering (Page(s): 198-207)

12. Exploiting Internal Parallelism of Flash-based SSDs

IEEE XploreIEEE Periodicals2010-01-013IEEE Computer Architecture Letters

13. IEEE Standard for Scalable Storage Interface

IEEE XploreIEEE Standards2006-03-223IEEE Std 1285-2005 (Page(s): 1-140)

14. A Format for a Graphical Communications Protocol

IEEE XploreIEEE Periodicals1986-03-013IEEE Computer Graphics and Applications

15. POSIX and the Verification Grand Challenge: A Roadmap

IEEE XploreIEEE Conferences2008-03-01313th IEEE International Conference on Engineering of Complex Computer Systems (iceccs 2008) (Page(s): 153-162)

16. An overview of logic architectures inside flash memory devices

IEEE XploreIEEE Periodicals2003-04-013Proceedings of the IEEE

17. SAS 9.1.3 Language Reference Concepts

IPCOM000243217DLegacy Journals2004-12-313SAS Institute Inc.

18. SAS 9.1 Language Reference Concepts

IPCOM000243169DLegacy Journals2004-12-313SAS Institute Inc.

19. IEEE Standard for a Smart Transducer Interface for Sensors and Actuators - Transducer to Microprocessor Communication Protocols and Transducer Electronic Data Sheet (TEDS) Formats

IEEE XploreIEEE Standards1998-09-253IEEE Std 1451.2-1997 (Page(s): 1-120)

20. ISO/IEC/IEEE International Standard for Information technology -- Smart transducer interface for sensors and actuators -- Part 2: Transducer to microprocessor communication protocols and Transducer Electronic Data Sheet (TEDS) formats

IEEE XploreIEEE Standards2010-05-153ISO/IEC/IEEE 21451-2:2010(E) (Page(s): 1-130)

21. Detecting Solid-State Disk Geometry for Write Pattern Optimization

IEEE XploreIEEE Conferences2011-08-0132011 IEEE 17th International Conference on Embedded and Real-Time Computing Systems and Applications

SAMSUNG EX1002, Page 92 of 1031

PTO/SB/30EFS (01-22)
Approved for use through 05/31/2024. OMB 0651-0031
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 contains a valid OMB control number.

REQUEST FOR CONTINUED EXAMINATION(RCE)TRANSMITTAL
(Submitted Only via EFS-Web)

Application Number	17931506	Filing Date	2022-09-12	Docket Number (if applicable)	RMMS.003C12	Art Unit	2136
First Named Inventor	Andrey V. Kuzmin			Examiner Name	Bataille, Pierre Miche		

This is a Request for Continued Examination (RCE) under 37 CFR 1.114 of the above-identified application.
Request for Continued Examination (RCE) practice under 37 CFR 1.114 does not apply to any utility or plant application filed prior to June 8, 1995, or to any design application. The Instruction Sheet for this form is located at WWW.USPTO.GOV

SUBMISSION REQUIRED UNDER 37 CFR 1.114

Note: If the RCE is proper, any previously filed unentered amendments and amendments enclosed with the RCE will be entered in the order in which they were filed unless applicant instructs otherwise. If applicant does not wish to have any previously filed unentered amendment(s) entered, applicant must request non-entry of such amendment(s).

☐ Previously submitted. If a final Office action is outstanding, any amendments filed after the final Office action may be considered as a submission even if this box is not checked.

- ☐ Consider the arguments in the Appeal Brief or Reply Brief previously filed on _____
- ☐ Other _____

☒ Enclosed

- ☐ Amendment/Reply
- ☒ Information Disclosure Statement (IDS)
- ☐ Affidavit(s)/ Declaration(s)
- ☐ Other _____

MISCELLANEOUS

☐ Suspension of action on the above-identified application is requested under 37 CFR 1.103(c) for a period of months _____
(Period of suspension shall not exceed 3 months; Fee under 37 CFR 1.17(i) required)

☐ Other _____

FEES

☒ **The RCE fee under 37 CFR 1.17(e) is required by 37 CFR 1.114 when the RCE is filed.**
The Director is hereby authorized to charge any underpayment of fees, or credit any overpayments, to Deposit Account No 111410

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT REQUIRED

☒ Patent Practitioner Signature

☐ Applicant Signature

PTO/SB/30EFS (01-22)
Approved for use through 05/31/2024. OMB 0651-0031
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 contains a valid OMB control number.

Signature of Registered U.S. Patent Practitioner			
Signature	/Aaron M. Davis/	Date (YYYY-MM-DD)	2023-03-23
Name	Aaron M. Davis	Registration Number	62027

This collection of information is required by 37 CFR 1.114. 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.

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

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

INFORMATION DISCLOSURE STATEMENT

First Inventor: Kuzmin, Andrey V.	**Conf. No.:** 3894
App. No.: 17/931,506	**Filed:** September 12, 2022
Examiner: Bataille, Pierre Miche	**Art Unit:** 2136
Title: STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS	

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

References and Listing

Pursuant to 37 CFR 1.56, an Information Disclosure Statement listing references is provided herewith. Copies of any listed foreign references are being submitted.

Pursuant to 37 CFR 1.97(g) and (h), Applicant makes no representation that the information is considered to be material to patentability. Additionally, inclusion on this list is not an admission that any of the cited documents are prior art in this application. Further, Applicant makes no representation regarding the completeness of this list, or that better art does not exist.

No Disclaimers

To the extent that anything in the Information Disclosure Statement or the listed references could be construed as a disclaimer of any subject matter supported by the present application, Applicant hereby rescinds and retracts such disclaimer.

Other Applications of Assignee

Applicant wishes to draw the Examiner's attention to, and encourages the Examiner to review, the following co-owned patents and/or applications and their existing and ongoing prosecution history:

Application No.	Filing Date	Patent No.
13/767,723	2/14/2013	9,652,376
14/047,193	10/7/2013	9,229,854
14/951,708	11/25/2015	9,519,578
15/346,641	11/8/2016	9,710,377
15/621,888	6/13/2017	10,445,229
16/570,922	9/13/2019	11,118,457
16/751,925	1/24/2020	11,314,636
16/783,083	2/5/2020	11,334,479
16/794,805	2/19/2020	
16/833,547	3/28/2020	11,216,365
17/563,481	12/28/2021	11,487,657
17/563,467	12/28/2021	11,487,656
17/571,394	1/7/2022	
17/877,881	7/29/2022	
17/931,516	9/12/2022	
17/952,964	9/23/2022	
17/952,230	9/24/2022	

SAMSUNG EX1002, Page 97 of 1031

Applicant notes that cited references, office actions, responses, and notices of allowance currently exist or will exist with reference to the above-referenced matters. Applicant also understands that the Examiner has access to sophisticated Patent Office computing systems that provide ready access to the full file histories of these matters including, for example, specifications, drawings, pending claims, cited art, office actions, responses, declarations, and notices of allowance. Rather than submit copies of these file histories, Applicant respectfully requests that the Examiner continue to review these file histories online for past, current, and future information about these matters that may be relevant to examination of the present application. With respect to parent applications of the present application, Applicant understands that, in accordance with MPEP 609, the Examiner of the present application will consider information which has been considered by the Patent Office in the parent applications. If the Examiner cannot readily access these file histories and information, Applicant would be pleased to provide any portion of any of the file histories or information at any time upon specific Examiner request.

Timing of Disclosure

This Information Disclosure Statement is being filed with a Request for Continued Examination, and no additional fee is believed to be required. The Commissioner is hereby authorized to charge any additional fees which may be required or to credit any overpayment to Account No. 11-1410.

Respectfully submitted,

KNOBBE, MARTENS, OLSON & BEAR, LLP

Dated: March 23, 2023

By: /Aaron M. Davis/

Aaron M. Davis
Registration No. 62,027
Registered Practitioner
(858) 707-4000

SAMSUNG EX1002, Page 98 of 1031

INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Application No.	17/931506
	Filing Date	September 12, 2022
	First Named Inventor	Kuzmin, Andrey V.
	Art Unit	2136
(Multiple sheets used when necessary)	Examiner	Bataille, Pierre Miche
SHEET 1 OF 1	Attorney Docket No.	RMMS.003C12

U.S. PATENT DOCUMENTS					
Examiner Initials	Cite No.	Document Number *Number - Kind Code (if known)* Example: 1,234,567 B1	Publication Date MM-DD-YYYY	Name	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear
	1	9,489,296	11-08-2016	Tomlin et al.	
	2	11,416,413	08-16-2022	Lercari et al.	
	3	11,449,436	09-20-2022	Lercari et al.	
	4	11,537,528	12-27-2022	Lercari et al.	
	5	11,537,529	12-27-2022	Lercari et al.	
	6	11,544,200	01-03-2023	Lercari et al.	
	7	2004/0083335	04-29-2004	Gonzalez	
	8	2005/0141313	06-30-2005	Gorobets et al.	
	9	2010/0026773	02-04-2010	Schultz et al.	
	10	2010/0077136	03-25-2010	Ware et al.	
	11	2010/0250837	09-30-2010	Schmidberger et al.	
	12	2011/0082967	04-07-2011	Deshkar et al.	
	13	2011/0161560	06-30-2011	Hutchison et al.	
	14	2012/0260150	10-11-2012	Cideciyan et al.	
	15	2013/0019057	01-17-2013	Stephens	
	16	2013/0024460	01-24-2013	Peterson et al.	
	17	2013/0122781	05-16-2013	Rutten et al.	

FOREIGN PATENT DOCUMENTS						
Examiner Initials	Cite No.	Foreign Patent Document *Country Code-Number-Kind Code* Example: JP 1234567 A1	Publication Date MM-DD-YYYY	Name	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear	**T[1]**
	18	WO 2011044154 A1	04-14-2011	Deshkar et al.		
	19	WO 2011055407 A1	05-12-2011	Takada et al.		

NON PATENT LITERATURE DOCUMENTS			
Examiner Initials	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	**T[1]**

Examiner Signature	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 next communication to applicant.	

T[1] - Place a check mark in this area when an English language Translation is attached.

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC PAYMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/931,506	03/23/2023 05:03:58 PM ET	RMMS.003C12

Title of Invention

STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	3894	FILED BY	Tayler Amerine
PATENT CENTER #	61794440	AUTHORIZED BY	Aaron Davis
CUSTOMER #	20995	FILING DATE	09/12/2022
CORRESPONDENCE ADDRESS	-	FIRST NAMED INVENTOR	Andrey V. Kuzmin

Payment Information

PAYMENT METHOD	PAYMENT TRANSACTION ID	PAYMENT AUTHORIZED BY
CARD / 1990	E20233MH04575508	Amy Nguyen

PRE-AUTHORIZED ACCOUNT	PRE-AUTHORIZED CATEGORY
111410	37 CFR 1.16 (National application filing, search, and examination fees); 37 CFR 1.17 (Patent application and reexamination processing fees)

FEE CODE	DESCRIPTION	ITEM PRICE($)	QUANTITY	ITEM TOTAL($)
2801	REQUEST FOR CONTINUED EXAMINATION (RCE) - 1ST REQUEST (SEE 37 CFR 1.114)	544.00	1	544.00
			TOTAL AMOUNT:	$544.00

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 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.

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC ACKNOWLEDGEMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/931,506	**03/23/2023 05:03:58 PM ET**	**RMMS.003C12**

Title of Invention

STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	3894	FILED BY	Tayler Amerine
PATENT CENTER #	61794440	FILING DATE	09/12/2022
CUSTOMER #	20995	FIRST NAMED INVENTOR	Andrey V. Kuzmin
CORRESPONDENCE ADDRESS	-	AUTHORIZED BY	Aaron Davis

Documents

TOTAL DOCUMENTS: 5

DOCUMENT		PAGES	DESCRIPTION	SIZE (KB)
003C12_RCE.pdf		3	Request for Continued Examination (RCE)	84 KB
Warning: This is not a USPTO supplied RCE fillable form. Data in the form cannot be automatically loaded to other USPTO systems.				
003C12_IDS.pdf		4	-	52 KB
003C12_IDS-TRAN.LET.pdf	(1-1)	1	Transmittal Letter	18 KB
003C12_IDS-IDS.pdf	(2-4)	3	Information Disclosure Statement (IDS) Form (SB08)	44 KB
FOR_WO2011044154A1.pdf		40	Foreign Reference	3405 KB

FOR_WO2011055407A1.pdf	47	Foreign Reference	5107 KB

Digest

DOCUMENT	MESSAGE DIGEST(SHA-512)
003C12_RCE.pdf	B387BB605A18C96D797B689688F2F79A7D397EA917DF81C594EFD20827F914FB2F230F2F60C59B712AA80CC40E8EB71144C08BF09A18F949F7F113963F2478DF
003C12_IDS.pdf	98724648E4E356B2EFDE79E83A5FFCB54CD45AA6486D51B0FB19BE81D1CE64FAFAB710199EAC961BBBAA0FA03414575 8B354C721919DFB18CA661C9E8703568C
003C12_IDS-TRAN.LET.pdf	218B5804DB7326FCB2C4AC1BD32E42A30AB95B99E1BC9B897908E8F1800C8DDF60EE0E54781C258BC545E22E8AD7AFD54E128C53FB8F9AF0D0DA36AB18AA6260
003C12_IDS-IDS.pdf	CF3B68A7314BE284125C29B00773292CB6215D965CC7DA463026F952068BF681A56024BA77F56FE504C4C87223BD7CAC579B82C4D798E60E3407349D7019576C
FOR_WO2011044154A1.pdf	35B794CA4F79F2DF47D3BCB4670B7A869FEB0453CB24E1C6D1B9B31A74A9488102D40A3311106383EA82091024BB902194F262C4C8E0EA5FFE80EA75ABAC36F9
FOR_WO2011055407A1.pdf	1D77FDC5E6F084B63C46030B8C59014A82F4AA589EE4ED1F3E23CE7A26C5CA0745B4980E5AC4EEFF65D9CB6C8E6E3090CDC04C2F1A2E30770A4E73D9E05414EB

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 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.

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
14 April 2011 (14.04.2011)

PCT

(10) International Publication Number
WO 2011/044154 A1

(51) International Patent Classification:
G06F 12/02 (2006.01) *G06F 12/08* (2006.01)

(21) International Application Number:
PCT/US2010/051514

(22) International Filing Date:
5 October 2010 (05.10.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/248,845 5 October 2009 (05.10.2009) US

(71) Applicant *(for all designated States except US)*: **MARVELL SEMICONDUCTOR, INC.** [US/US]; 5488 Marvell Lane, Santa Clara, California 95054 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **DESHKAR, Shekhar S.** [IN/IN]; Flat No. 11, Bandhangad Apartments, D.P. Road, Aundh, Pune 411007 (IN). **KARMARKAR, Sandeep** [IN/IN]; 401 Sairang Near Paranjape School, Kothrud, Pune 411038 (IN). **PRUTHI, Arvind** [US/US]; 741 Pollard Road, Apt. 1, Los Gatos, California 95032 (US). **JOHRI, Ram Kishore** [US/US]; 1526 Miller Ave., San Jose, California 95129 (US).

(74) Agents: **HUNTER, William E.** et al.; Fish & Richardson P.C., P. O. Box 1022, Minneapolis, Minnesota 55440-1022 (US).

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

[Continued on next page]

(54) Title: DATA CACHING IN NON-VOLATILE MEMORY

(57) Abstract: Methods, systems, and apparatus, including computer programs encoded on a computer storage medium, can perform data caching. In some implementations, a method and system include receiving information that includes a logical address, allocating a physical page in a non-volatile memory structure, mapping the logical address to a physical address of the physical page, and writing, based on the physical address, data to the non-volatile memory structure to cache information associated with the logical address. The logical address can include an identifier of a data storage device and a logical page number.

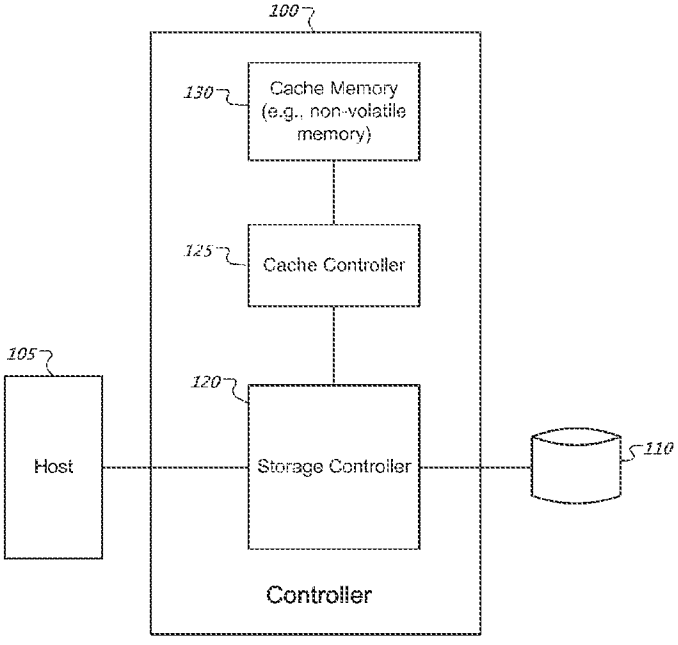


FIG. 1

WO 2011/044154 A1

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

---- *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

DATA CACHING IN NON-VOLATILE MEMORY

CROSS REFERENCE TO RELATED APPLICATION

[0001] This application claims the benefit of the priority of U.S. Provisional Application Serial No. 61/248,845, filed October 5, 2009 and entitled "Flash Based Cache with a Cache Aware Flash Translation Layer." The content of the above application is incorporated by reference in its entirety.

BACKGROUND

[0002] The present disclosure describes systems and techniques relating to data caching.

[0003] A host system can communicate with a storage system to read and write data. A host system can include a computer system, such as a personal computer, a consumer electronics device, or a server system. A storage system can include one or more data storage devices such as a hard disk drive (HDD) or a solid state drive (SSD) communicating with the host system. In some implementations, a host system includes the storage system, and in other implementations, the storage system is external to the host system. A storage system can perform data caching to decrease average access latency times.

SUMMARY

[0004] The present disclosure includes systems and techniques relating to data caching. According to an aspect of the described systems and techniques, a system includes a host system, a data storage device, a non-volatile memory structure that is configured to cache data with respect to the data storage device, and a controller configured to operate devices such as the data storage device. The controller can receive a logical address from the host system, allocate a physical page in the non-volatile memory structure, store a mapping from the logical address to a physical address of the physical page, receive data associated with the

SAMSUNG EX1002, Page 107 of 1031

logical address, and write, based on the physical address, the data associated with the logical address to the non-volatile memory structure to cache the data associated with the logical address. The mapping can indicate a cached status of the data associated with the logical address.

[0005] In some implementations, a controller is configured to receive the data associated with the logical address from one of the host system or the data storage device. In some implementations, a controller is configured to invalidate the mapping based on the data associated with the logical address being written to a different physical page. A controller can perform an erasure of a physical page associated with an invalidated mapping to make the physical page available for re-allocation. In some implementations, a controller is configured to invalidate an allocated physical page based a logical address indicated by an invalidate command sequence. In some implementations, a logical address includes an identifier of a data storage device and a logical page number. A system can include two or more data storage devices. The non-volatile memory structure can be configured to cache data with respect to the two or more data storage devices. A mapping can include a data storage device identifier that identifies at least one of the two or more data storage devices. In some implementations, the controller includes a storage controller and a cache controller, and the cache controller is communicatively coupled with the non-volatile memory structure. A non-volatile memory structure can include one or more flash memory structures. A data storage device can include one or more hard disk drives. A system can include caching devices that include non-volatile memory structures. In some implementations, the controller is configured to select a caching device based on the logical address.

[0006] According to another aspect of the described systems and techniques, a method includes receiving, from a host system, information that includes a logical address. The method includes receiving data associated with the logical address from one of the host

SAMSUNG EX1002, Page 108 of 1031

system or a data storage device. The method includes allocating a physical page in a non-volatile memory structure. In some implementations, the non-volatile memory structure is configured to cache data with respect to at least one data storage device. The method includes storing a mapping from the logical address to a physical address of the physical page. The method includes writing, based on the physical address, the data associated with the logical address to the non-volatile memory structure to cache the data associated with the logical address. The mapping can indicate a cached status of the data associated with the logical address.

[0007] In some implementations, a method includes invalidating the mapping based on the data associated with the logical address being written to a different physical page and performing an erasure of the physical page associated with the invalidated mapping to make the physical page available for re-allocation. Receiving data associated with the logical address can include reading data from the data storage device based on the logical address. In some implementations, a method includes invalidating an allocated physical page based a logical address indicated by an invalidate command sequence.

[0008] The described systems and techniques can be implemented in electronic circuitry, computer hardware, firmware, software, or in combinations of them, such as the structural means disclosed in this specification and structural equivalents thereof. This can include at least one computer-readable medium embodying a program operable to cause one or more data processing apparatus to perform operations described. Thus, program implementations can be realized from a disclosed method, system, or apparatus, and apparatus implementations can be realized from a disclosed system, computer-readable medium, or method. Similarly, method implementations can be realized from a disclosed system, computer-readable medium, or apparatus, and system implementations can be realized from a disclosed method, computer-readable medium, or apparatus.

SAMSUNG EX1002, Page 109 of 1031

[0009] An apparatus can include a non-volatile memory structure, circuitry to receive a command sequence that includes a logical address, circuitry to receive data associated with the logical address, circuitry to allocate a physical page in the non-volatile memory structure, circuitry to store a mapping from the logical address to a physical address associated with the physical page, and circuitry to write, based on the physical address, the data associated with the logical address to the non-volatile memory structure to cache the data associated with the logical address. The mapping can indicate a cached status of the data associated with the logical address. The logical address can include information that identifies a data storage device and a logical page number.

[0010] In some implementations, an apparatus can include circuitry to receive a read command sequence that includes a logical address, circuitry to identify a physical address associated with the logical address of the read command sequence, and circuitry to read, based on the identified physical address, data from the non-volatile memory structure to provide cached data associated with the logical address of the read command sequence. An apparatus can include circuitry to invalidate an allocated physical page based a logical address indicated by an invalidate command sequence.

[0011] The described systems and techniques can result in a reduced latency storage system. For example, using a mapping that relates a logical address to a physical address of a page on a non-volatile memory can reduce a number of lookups required to read data from a cache data store. In addition, rather than maintaining multiple separate lookup tables, a single lookup table can be employed to more efficiently use memory in a cache data store. This approach to data caching in a storage system can result in reduced latency.

[0012] Details of one or more implementations are set forth in the accompanying drawings and the description below. Other features, objects, and advantages may be apparent from the description and drawings, and from the claims.

SAMSUNG EX1002, Page 110 of 1031

DRAWING DESCRIPTIONS

[0013] FIG. 1 shows an example of a storage architecture.

[0014] FIG. 2 shows an example of a cache controller architecture.

[0015] FIG. 3 shows an example of a block and page layout of a non-volatile memory.

[0016] FIG. 4 shows an example of a cache based storage process.

[0017] FIG. 5 shows an example of a mapping table layout.

[0018] FIG. 6 shows an example of a cache aware flash translation layer process.

[0019] FIGS. 7A and 7B show examples of different cache processes.

[0020] FIG. 8 shows an example of a storage system with multiple flash memory structures.

[0021] FIG. 9 shows an example of a storage process.

[0022] Like reference symbols in the various drawings indicate like elements.

DETAILED DESCRIPTION

[0023] FIG. 1 shows an example of a storage architecture. A host 105 can access one or more storage devices 110 via a controller 100. A controller 100 can include a storage controller 120, a cache controller 125, and one or more memories such as a cache memory 130. In some implementations, a storage device 110 includes one or more storage elements that are separately addressable by the controller 100.

[0024] A host 105 can send a command sequence such as a write command sequence or a read command sequence to the storage controller 120. A command sequence can include a logical address. In some implementations, a logical address includes a storage device target identifier that is associated with a storage device 110. In some implementations, a command sequence can include a command identifier, a logical address, and a length. In some implementations, a host 105 communicates with a storage controller 120 via an interface such

SAMSUNG EX1002, Page 111 of 1031

as a Peripheral Component Interconnect (PCI), Serial Advanced Technology Attachment (SATA), Small Computer System Interface (SCSI), Serial Attached SCSI (SAS), Ethernet, or Universal Serial Bus (USB).

[0025] A storage controller 120 can communicate with a cache controller 125 to cache data received from a storage device 110, cache data received from the host 105, or both. The cache controller 125 can communicate with a cache memory 130, which can include non-volatile memory such as flash memory. In some implementations, the cache controller 125 includes a flash memory controller to control flash memory in the cache memory 130. For example, a storage controller 120 can read data from a storage device 110 and store at least a portion of the read data in a cache memory 130 controlled by a cache controller 125. In another example, a storage controller 120 can receive data from a host 105 for writing to a storage device 110. A storage controller 120 can store write data, received from the host 105, in a cache memory 130 controlled by a cache controller 125. In some implementations, a storage controller 120 stores host write data on a cache memory 130 and is not required to immediately store the host write data on a storage device 110. In some implementations, a storage controller 120 stores host write data on a cache memory 130 and on a storage device 110.

[0026] A host 105 can include one or more processors, memory, and input/output (I/O) connections. In some implementations, a host 105 includes a storage controller 120. For example, a system-on-a-chip (SOC) can include a general purpose processor and one or more controllers (e.g., storage controller, cache controller, or a combination thereof) to access external devices such as hard disk drives (HDDs) or solid-state drives (SSDs). A SSD can include one or more non-volatile memory structures such as a non-volatile memory integrated circuit, e.g., flash chip. A cache controller 125 can include one or more processors and one or more non-volatile memory. In some implementations, a storage controller 120

SAMSUNG EX1002, Page 112 of 1031

includes one or more cache controllers 125. In some implementations, a storage controller 120 is integrated with a cache controller 125 and one or more storage devices 110.

[0027] FIG. 2 shows an example of a cache controller architecture. A cache controller 205 can cache data for a storage controller. Caching data can increase performance, e.g., faster read times and faster write times. A cache controller 205 can communicate with one or more non-volatile memory chips such as flash chips 210 to store or retrieve cache data. A cache controller 205 can communicate with a controller memory 215, which stores cache metadata such as address mappings for cached data in the flash chips 210. In some implementations, a controller memory 215 is integrated into a memory region of a flash chip 210, e.g., a portion of memory in a flash chip is configured as a memory for the cache controller 205.

[0028] The cache controller 205 can store information such as cache metadata on the controller memory 215. For example, the cache controller 205 can store one or more mappings in a mapping table 220 to facilitate caching operations. A mapping, stored on the mapping table 220, identifies a relationship between a logical address (LA) and a physical address (PA) corresponding to a physical page stored on at least one of the flash chips 210. A logical address can include a logical page number (LPN) associated with a data location on a storage device. In some implementations, a LPN can be referred to as a logical block address (LBA). In some implementations, a LPN can include an offset and a length. A physical address can include a block identifier and a page identifier.

[0029] In some implementations, a mapping table 220 can include logical addresses that include a LPN and a storage device target identifier that is associated with a data storage device. A storage device target identifier can differentiate identical LPNs. For example, two data storages devices can store different data using identical LPNs. A controller can use a

SAMSUNG EX1002, Page 113 of 1031

storage device target identifier to retrieve data associated with a LPN from a corresponding storage device.

[0030] The cache controller 205 can receive a command sequence from a storage controller to retrieve data from cache. The command sequence can identify a logical address. If the cache controller 205 has a cache hit, e.g., the command sequence's logical address is found in the mapping table 220, then the cache controller 205 uses the corresponding physical address to retrieve data from a flash chip 210 corresponding to the physical address. The cache controller 205 can send the retrieved data to the storage controller.

[0031] If the cache controller 205 has a cache miss, e.g., a command's logical address is not found in the mapping table 220, then the cache controller 205 can report the miss to a storage controller. In some implementations, the cache controller 205 can report the miss to a program such as firmware. However, the cache controller 205 can allocate a physical page in a flash chip 210 to store a mapping in the mapping table 220 that identifies a relationship between the command's logical address and a physical address of the allocated physical page. After the storage controller services the cache miss, e.g., retrieves data corresponding to the logical address from a storage device, the cache controller 205 can receive the data and store the data in one or more of the flash chips 210 for a future read access.

[0032] In another example, the cache controller 205 receives a command to write data to the cache. The command identifies a logical address. If the cache controller 205 has a cache hit, the cache controller 205 uses a corresponding physical address to write data to a flash chip 210 corresponding to the physical address. In some implementations, a new location is required for a future write to the same physical page. Therefore, the cache controller 205 can allocate a new physical page and update the mapping table 220 to reflect the new physical page assignment for the logical address. In some implementations, the cache controller 205 can mark the old physical page invalid.

SAMSUNG EX1002, Page 114 of 1031

[0033] FIG. 3 shows an example of a block and page layout of a non-volatile memory. A non-volatile memory layout 300 includes one or more blocks 305. A block 305 includes one or more pages 310. For example, a block can be 256 KB in size and a page can be 4 KB in size, and accordingly, a block can include 64 pages. In another example, a block can be 512 KB in size and a page can be 4 KB in size, hence, a block can include 128 pages.

[0034] A storage controller can receive information, such as a command sequence from a host system. A command sequence can include a logical address. For a write command, a command sequence can include data. In some implementations, a logical address includes an identifier of a data storage device and a logical page number. A storage controller can run a cache based storage process. In some implementations, a storage controller operates a cache controller to run at least a portion of a cache based storage process. For example, the storage controller can send a command sequence to a cache controller based upon a command sequence received from the host system.

[0035] FIG. 4 shows an example of a cache based storage process. In some implementations, a cache based storage process includes operations for a storage controller and operations for a cache controller that is in communication with the storage controller. In some implementations, the process runs on a controller such as a storage controller or an integrated storage and cache controller. At 405, a cache based storage process can receive a command sequence that includes a logical address from a host system. In some cases, a command sequence indicates that data are to be retrieved for the host system. In some cases, a command sequence indicates a write operation and includes host write data.

[0036] At 410, the cache based storage process can access a mapping table to determine if a mapping associated with the logical address exists. For example, the process can perform a table lookup based on the logical address. In some implementations, the process uses a hash table to determine whether a mapping associated with the logical address exists. In some

SAMSUNG EX1002, Page 115 of 1031

implementations, a cache controller stores a table in a controller memory and uses the table to perform a lookup based on a logical address.

[0037] At 415, if a mapping does not exist, then the process, at 420, can allocate a physical page in a non-volatile memory structure. The process, at 425, can generate a mapping to relate the logical address with a physical address of the physical page. In some implementations, a physical address of a physical page is a starting address of the physical page. At 430, the process can store the mapping in the mapping table. The mapping can be indicative of a cached status of the data associated with the logical address. For example, the existence of a valid mapping table entry can indicate the presence of cached data. At 435, the process can write, based on the physical address, data to the non-volatile memory structure to cache information associated with the logical address. For example, a process can receive a command sequence that includes host write data. The process can cache host write data in the non-volatile memory structure.

[0038] At 415, if the mapping does exist, then the process, at 440, can use the existing mapping to handle a data operation with the non-volatile memory structure. Various examples of data operations include a write operation and a read operation. In some implementations, the process uses the existing mapping to read data from the non-volatile memory structure. In some implementations, the process uses the existing mapping to write data to the non-volatile memory structure. In some implementations, using the existing mapping to write data can include invalidating the existing mapping and creating a new mapping to write data to a different physical location in the non-volatile memory structure.

[0039] FIG. 5 shows an example of a mapping table layout. A mapping table 500 layout is arranged to store one or more entries 505. A mapping table 500 is partitioned into logical addresses 510 and respective physical addresses 515. For example, an entry 505 includes a logical address 510 and a physical address 515. In some implementations, a logical address

SAMSUNG EX1002, Page 116 of 1031

includes an identifier such as a target device identifier (e.g., TGT 1, TGT 2) that identifies a target device. For example, a logical address can include a target device identifier and a LPN. With respect to a read command sequence, a controller can read, using a LPN, a page from a device corresponding to a target device identifier included in the sequence. The controller can cache the page. Caching the page can include creating a mapping, in the mapping table 500, that includes the target device identifier and the LPN. In some implementations, an entry 505 includes an active flag to indicate whether the entry 505 is in use. An entry 505 can include a valid flag to indicate whether a physical page associated with a physical address 515 is valid or invalid. For example, an entry 505 is marked as being in use and the valid flag is marked as invalid. An entry 505 that is marked as being in use and the valid flag is marked as valid can indicate a cache status of true.

[0040] A storage system can include one or more non-volatile memories configured to cache information flowing to a storage device or coming from a storage device. Non-volatile memories such as flash memory can be electronically erased and reprogrammed. In some implementations, flash memory can include NAND logic.

[0041] In some implementations, flash memory operates based on erase blocks and pages. For example, a unit of access is a page, whereas a unit of an erase operation is an erase-block. Sizes of blocks and pages can vary with different flash devices. For example, in relation to a flash device included in a large-block flash SSD, an erase block size is between 256KB to 512KB in size and a page is 4KB in size. However, other sizes are possible.

[0042] In some implementations, a flash block, which already contains data, cannot be re-written unless the block is explicitly erased. An erase operation can be time expensive, e.g., can take several milliseconds. In some implementations, for large block NAND devices, the pages within an erase block cannot be written randomly. For example, an erase block includes 100 pages numbered from 1 to 100. In some implementations, if page 20 within the

SAMSUNG EX1002, Page 117 of 1031

erase block is written, then pages 1-20 cannot be written to unless the entire block is erased. However, pages after page 20 are write-able.

[0043] In some flash memory designs, a block can be erased only a finite number of times before it becomes unusable due to one or more physical characteristics of flash memory. A flash memory device can use wear-leveling algorithms to distribute a load on multiple flash chips. A flash memory device can use wear-leveling algorithms to make sure that usage of erase blocks within a flash memory is rotated such that no single flash erase block gets burdened with too many write cycles.

[0044] An application may write and subsequently rewrite data to the same logical page. The logical page can be mapped to a physical page stored in a flash memory. Rewriting a page to the same location can require a flash device to read the entire erase block into memory, erase the entire block, rewrite data from memory to pages prior to the page that needs to be updated, update the relevant page, and rewrite the remaining pages in the block to the flash. The operations for rewriting a page to the same location can be expensive in terms of time. Further, there is a possibility of permanent data loss if the flash device is powered off before the rewriting process is completed. However, rewriting in place may not ensure a uniform wear of flash blocks.

[0045] A flash device can re-write a page to a different location to increase operational latency and provide a more distributed wear-leveling of the flash memory. For example, a flash device can mark the page as invalid and write the updated page to a different location. In some implementations, a flash device can use an abstraction layer of logical pages on top of the physical space provided by a flash memory structure. The layer is called as a flash translation layer (FTL). In some implementations, commands sent to the flash devices can use logical addresses to refer to data. In some implementations, a command includes a LPN. The FTL can assign a physical page number to a logical page number using logic internal to

SAMSUNG EX1002, Page 118 of 1031

the FTL. For re-writing, the FTL can be configured to re-write pages to a different location. Hence, a flash may have multiple physical pages that are associated with the same logical page of which only one physical page is deemed valid and the rest are deemed invalid. A flash device can use FTL metadata to track which physical locations contain valid data.

[0046] FTL metadata can include a table that stores a mapping table of LPNs and respective physical page locations. In some implementations, a LPN is a logical page address exported via interface such as SAS, SATA, or PCI. The mapping table can maintain the latest physical location that has valid contents for a corresponding LPN. FTL metadata can include a table that describes which physical pages have valid data.

[0047] A flash device can perform garbage collection to reclaim invalidated pages such that they can be reused for a future write operation. In some implementations, FTL logic requires a free block to process an update to a logical page. In some implementations, a processor within the flash controller runs a garbage collection process in the background. A garbage collection process can search for erase blocks and run a de-fragmentation operation before making the page available for write. In some implementations, the process searches for blocks based on a number of invalidated pages contained therein. A de-fragmentation process can collect the valid pages from various blocks in memory, rewrite them into a new erase block, update the metadata mapping, and erase the old blocks. The erased blocks are now available for a new write. A flash device can perform on-demand garbage collection in the event that no free physical pages exist to satisfy a write operation. However, to reduce the likelihood of on-demand garbage collection, a flash controller can declare a lesser amount of storage space than what is physically available.

[0048] A storage system can include a flash device that is configured as a caching device. A caching device can include a cache-aware FTL. Rather than supporting a direct mapping between LPNs and physical storage space, such as when a flash device is used as a target-

SAMSUNG EX1002, Page 119 of 1031

device, a caching device can provide caching for multiple target devices with a logical-page range independent of the actual physical space provided by the flash device.

[0049] In some implementations, a caching device can provide commands such as GET and SET commands to manipulate the device. Such commands can accept a target device identifier, e.g., TGT-ID, in addition to a logical page address and transfer length fields. In some implementations, a logical address includes a TGT-ID and a logical page number. The TGT-ID can identify a target device associated with data sent or received via a command. A SET operation can include caching the supplied data, identified by offset and length, for a specified TGT-ID. Caching the supplied data can include storing the supplied data on a flash memory structure. A GET operation can include fetching the data identified by an offset and length specified in the GET call for a specified TGT-ID.

[0050] A cache aware FTL mapping table can include a mapping of an application specified logical address, such as <TGT-ID, LPN>, to a physical location description within a flash memory structure. A physical location description can identify a flash block and a page within the flash block. In some cases, an entry in a mapping table may include a logical address that does not have a corresponding valid physical page (e.g., a physical location description is marked as "invalid"). Such an entry can be eliminated from the mapping table.

[0051] A caching flash device can provide an interface to INVALIDATE a cache mapping. An INVALIDATE command can be used when an application no longer requires a page stored in the cache. A cache aware FTL can track physical page status, e.g., valid or invalid. A cache aware FTL can continue to do garbage collection to maintain a pool of erased blocks.

[0052] In response to a GET call, a cache aware FTL can directly lookup an application supplied logical address in the FTL's internal mappings to check if the data exists within the flash. If not, the FTL can send out an error message.

SAMSUNG EX1002, Page 120 of 1031

[0053] In response to a SET call, a cache aware FTL can check if a free page is available to accommodate an incoming write request indicated by the SET call. If a free page does exist, the FTL can transfer the contents of the data to the available page and create a new entry in the FTL's mapping table to indicate the mapping of this new block. The FTL can invalidate an old mapping, if such a mapping exists. If a page is unavailable, the FTL can wait for a garbage collection process to reclaim pages. In some implementations, if a page is unavailable, the FTL can return back with an error message.

[0054] In response to an INVALIDATE call, a cache aware FTL can lookup corresponding logical addresses in the FTL's mapping table, remove these mappings from the mapping table, and can mark corresponding physical pages invalid. A garbage collection process can reclaim pages marked invalid during an INVALIDATE call.

[0055] A caching flash device can support several target devices. While a caching flash device can theoretically provide caching for an unlimited amount of physical space, a notion of a ratio of caching space provided by a flash to the physical space for which the flash acts as a cache is useful to understand the device's performance. In some cases, a smaller ratio may lead to a smaller working set in cache and, hence, results in less effective caching.

[0056] FIG. 6 shows an example of a cache aware flash translation layer process. A caching device can run a cache aware flash translation layer process to handle incoming requests such as a SET call. A storage controller can send a SET call to a caching device to cache application data on the caching device. At 605, the process can receive, via a SET call, information including a logical address and application data. At 610, the process determines whether a mapping corresponding to the logical address exists. If the mapping exists, then, the process, at 615, marks the physical page that is identified by the mapping as invalid. At 620, the process deletes the mapping. In some implementations, the process marks an entry that contains the mapping as invalid.

SAMSUNG EX1002, Page 121 of 1031

[0057] At 625, the process determines whether a free physical page exists. If a free physical page does not exist, then the process runs a garbage collection process to free one or more physical pages at 630. If a free physical page does exist, then, at 635, the process allocates the physical page at 635. At 640, the process creates a new mapping to indicate a relationship between the logical address and the allocated physical page. At 645, the process transfers the application data received via the SET call to the allocated physical page on a flash memory structure.

[0058] FIGS. 7A and 7B show examples of different cache processes. The cache process 705 depicted by FIG. 7A includes two look-ups, whereas the cache process 710 depicted by FIG. 7B includes one look-up. A cache process that uses less look-ups can provide better performance over a cache process that uses more look-ups. Examples of better performance include reduced latency, reduced complexity, or both.

[0059] The cache process 705 depicted by FIG. 7A includes receiving a command that includes a logical address, e.g., <TGT-ID, LPN>. The process 705 includes performing a first look-up on the logical address to obtain a LPN of page in a cache memory such as a flash memory. The process 705 includes performing a second look up on the LPN to obtain a physical address of a physical page. The process 705 includes performing a read from or write to the physical page.

[0060] The cache process 710 depicted by FIG. 7B includes receiving a command that includes a logical address. The process 710 includes performing a look-up on the logical address to obtain a physical address of page in a cache memory such as a flash memory. The process 710 includes performing a read from or write to the physical page. The process 710 can access cache-aware FTL metadata that includes a mapping table associating logical addresses, e.g., <TGT-ID, LPN>, with physical addresses of physical pages.

SAMSUNG EX1002, Page 122 of 1031

[0061] A cache-aware FTL can respond to commands such as an INVALIDATE command. Responding to such a command can include removing one or more associated mappings (e.g., {<TGT-ID, LPN> <=> physical page}) from the FTL's mapping table. A mapping table may have multiple mappings that are associated with the same logical address as a result of previous page invalidations. Removing such mappings can increase flash memory space utilization and can decrease a garbage collector's processing load.

[0062] In some implementations, a cache controller uses flash memory as a write-back cache. The cache controller can reserve some space in the flash memory as a buffer space. This buffer space can be used to match up a higher incoming I/O rate with a lower cache replacement rate. In some implementations, a FTL can reserve space in the flash memory for a garbage collection process to allow the process to catch up with an incoming I/O data stream if such a scenario arises.

[0063] A cache-aware FTL can reserve less space when compared to a FTL that is not cache-aware. This is because as caching logic starts invalidating cache blocks, the corresponding physical pages also start becoming invalid, which hence become candidates for erasing. A garbage collector can directly use these invalid pages to produce pages for allocation. However, in some implementations, a cache-aware FTL is not required to reserve space.

[0064] In some implementations, a storage controller can use two or more caching devices to cache data from storage devices. A storage controller can distribute data caching across multiple caching devices. In some implementations, a storage controller can use a hash function that maps the tuple <TGT-ID, offset, length> to a cache device. The cache device that is identified by an output of the hash function can be used for operations (e.g., GET, SET, INVALIDATE) for that tuple.

SAMSUNG EX1002, Page 123 of 1031

[0065] FIG. 8 shows an example of a storage system with multiple flash memory structures. A storage system can include data storage devices 810a, 810b such as a HDD, SSD, or a combination thereof. A controller 815 can write and read data to the data storage devices 810a, 810b. A controller 815 can communicate with one or more caching devices 820. A caching device 820 includes memory to cache data such as one or more flash memory structures, e.g., one or more flash chips 845. The caching device 820 can include interface logic 825, a processor 830, a random access memory (RAM) 835, and a buffer manager 840

[0066] FIG. 9 shows an example of a storage process. At 905, a storage controller can receive a command sequence that includes a logical address (e.g., <TGT-ID, LPN>). At 910, a storage controller can select a caching device based on the logical address. Selecting a caching device can include applying a hash function to the logical address to produce an identifier of a caching device. At 915, the storage controller can send the logical address and associated information to the selected caching device to handle the command sequence. At 920, if there is a cache read miss, then at 925, the storage controller can read data associated with the LPN from the data storage device identified by the TGT-ID. At 930, the storage controller can operate the caching device to write the read data to a flash memory structure. At 935, the storage controller can send a response to the command sequence. For a cache read miss, the response can include data retrieved from the storage device.

[0067] At 920, if there is not a cache read miss, then the storage controller can send a response to the command sequence at 935. For a write command sequence, the response can include a status, e.g., write was successful.

[0068] A few embodiments have been described in detail above, and various modifications are possible. The disclosed subject matter, including the functional operations described in this specification, can be implemented in electronic circuitry, computer hardware, firmware, software, or in combinations of them, such as the structural means

SAMSUNG EX1002, Page 124 of 1031

disclosed in this specification and structural equivalents thereof, including potentially a program operable to cause one or more data processing apparatus to perform the operations described (such as a program encoded in a computer-readable medium, which can be a memory device, a storage device, a machine-readable storage substrate, or other physical, machine-readable medium, or a combination of one or more of them).

[0069] The term "data processing apparatus" encompasses all apparatus, devices, and machines for processing data, including by way of example a programmable processor, a computer, or multiple processors or computers. The apparatus can include, in addition to hardware, code that creates an execution environment for the computer program in question, e.g., code that constitutes processor firmware, a protocol stack, a database management system, an operating system, or a combination of one or more of them.

[0070] A program (also known as a computer program, software, software application, script, or code) can be written in any form of programming language, including compiled or interpreted languages, or declarative or procedural languages, and it can be deployed in any form, including as a stand alone program or as a module, component, subroutine, or other unit suitable for use in a computing environment. A program does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that holds other programs or data (e.g., one or more scripts stored in a markup language document), in a single file dedicated to the program in question, or in multiple coordinated files (e.g., files that store one or more modules, sub programs, or portions of code). A program can be deployed to be executed on one computer or on multiple computers that are located at one site or distributed across multiple sites and interconnected by a communication network.

[0071] While this specification contains many specifics, these should not be construed as limitations on the scope of what may be claimed, but rather as descriptions of features that may be specific to particular embodiments. Certain features that are described in this

SAMSUNG EX1002, Page 125 of 1031

specification in the context of separate embodiments can also be implemented in combination in a single embodiment. Conversely, various features that are described in the context of a single embodiment can also be implemented in multiple embodiments separately or in any suitable subcombination. Moreover, although features may be described above as acting in certain combinations and even initially claimed as such, one or more features from a claimed combination can in some cases be excised from the combination, and the claimed combination may be directed to a subcombination or variation of a subcombination.

[0072] Similarly, while operations are depicted in the drawings in a particular order, this should not be understood as requiring that such operations be performed in the particular order shown or in sequential order, or that all illustrated operations be performed, to achieve desirable results. In certain circumstances, multitasking and parallel processing may be advantageous. Moreover, the separation of various system components in the embodiments described above should not be understood as requiring such separation in all embodiments.

[0073] Other embodiments fall within the scope of the following claims.

SAMSUNG EX1002, Page 126 of 1031

CLAIMS

WHAT IS CLAIMED IS:

1. A system comprising:

a host system;

a data storage device;

a non-volatile memory structure that is configured to cache data with respect to the data storage device; and

a controller configured to (i) operate the data storage device, (ii) receive a logical address from the host system, (iii) allocate a physical page in the non-volatile memory structure, (iv) store a mapping from the logical address to a physical address of the physical page, (v) receive data associated with the logical address, and (vi) write, based on the physical address, the data associated with the logical address to the non-volatile memory structure to cache the data associated with the logical address, wherein the mapping indicates a cached status of the data associated with the logical address.

2. The system of claim 1, wherein the controller is configured to receive the data associated with the logical address from one of the host system or the data storage device.

3. The system of claim 1, wherein the controller is configured to (i) invalidate the mapping based on the data associated with the logical address being written to a different physical page and (ii) perform an erasure of the physical page associated with the invalidated mapping to make the physical page available for re-allocation.

4. The system of claim 1, wherein the controller is configured to invalidate an allocated physical page based a logical address indicated by an invalidate command sequence.

SAMSUNG EX1002, Page 127 of 1031

5. The system of claim 1, comprising:

two or more data storage devices including the data storage device,

wherein the non-volatile memory structure is configured to cache data with respect to the two or more data storage devices, and wherein the mapping includes a data storage device identifier that identifies at least one of the two or more data storage devices.

6. The system of claim 1, wherein the controller includes a storage controller and a cache controller, wherein the cache controller is communicatively coupled with the non-volatile memory structure.

7. The system of claim 1, wherein the non-volatile memory structure includes a flash memory structure, and wherein the data storage device includes a hard disk drive.

8. The system of claim 1, comprising:

a first caching device that includes the non-volatile memory structure; and

a second caching device,

wherein the controller is configured to select a caching device based on the logical address.

SAMSUNG EX1002, Page 128 of 1031

9. An apparatus, comprising:

a non-volatile memory structure;

circuitry to receive a command sequence that includes a logical address, wherein the logical address includes (i) information that identifies a data storage device and (ii) a logical page number;

circuitry to receive data associated with the logical address;

circuitry to allocate a physical page in the non-volatile memory structure;

circuitry to store a mapping from the logical address to a physical address associated with the physical page; and

circuitry to write, based on the physical address, the data associated with the logical address to the non-volatile memory structure to cache the data associated with the logical address, wherein the mapping indicates a cached status of the data associated with the logical address.

10. The apparatus of claim 9, comprising:

circuitry to receive a read command sequence that includes a logical address;

circuitry to identify a physical address associated with the logical address of the read command sequence; and

circuitry to read, based on the identified physical address, data from the non-volatile memory structure to provide cached data associated with the logical address of the read command sequence.

11. The apparatus of claim 9, comprising:

circuitry to invalidate an allocated physical page based a logical address indicated by an invalidate command sequence.

SAMSUNG EX1002, Page 129 of 1031

12. The apparatus of claim 9, wherein the non-volatile memory structure comprises a flash memory structure.

13. A method, comprising:

receiving, from a host system, information that includes a logical address;

receiving data associated with the logical address from one of the host system or a data storage device;

allocating a physical page in a non-volatile memory structure, wherein the non-volatile memory structure is configured to cache data with respect to at least one data storage device;

storing a mapping from the logical address to a physical address of the physical page; and

writing, based on the physical address, the data associated with the logical address to the non-volatile memory structure to cache the data associated with the logical address, wherein the mapping indicates a cached status of the data associated with the logical address.

14. The method of claim 13, comprising:

invalidating the mapping based on the data associated with the logical address being written to a different physical page; and

performing an erasure of the physical page associated with the invalidated mapping to make the physical page available for re-allocation.

15. The method of claim 13, wherein receiving data associated with the logical address comprises reading data from the data storage device based on the logical address.

SAMSUNG EX1002, Page 130 of 1031

16. The method of claim 13, comprising:

invalidating an allocated physical page based a logical address indicated by an invalidate command sequence.

17. The method of claim 13, wherein the non-volatile memory structure is configured to cache data with respect to two or more data storage devices, and wherein the logical address includes a target device identifier that identifies at least one of the two or more data storage devices.

18. The method of claim 13, comprising:

selecting a caching device from two or more caching devices based on the logical address, wherein writing data to the non-volatile memory structure comprises communicating with the selected caching device.

19. The method of claim 13, wherein selecting the caching device comprises applying a hash function to the logical address to produce an identifier of a caching device.

20. The method of claim 13, wherein writing data to the non-volatile memory structure comprises writing to a flash memory structure.

SAMSUNG EX1002, Page 131 of 1031

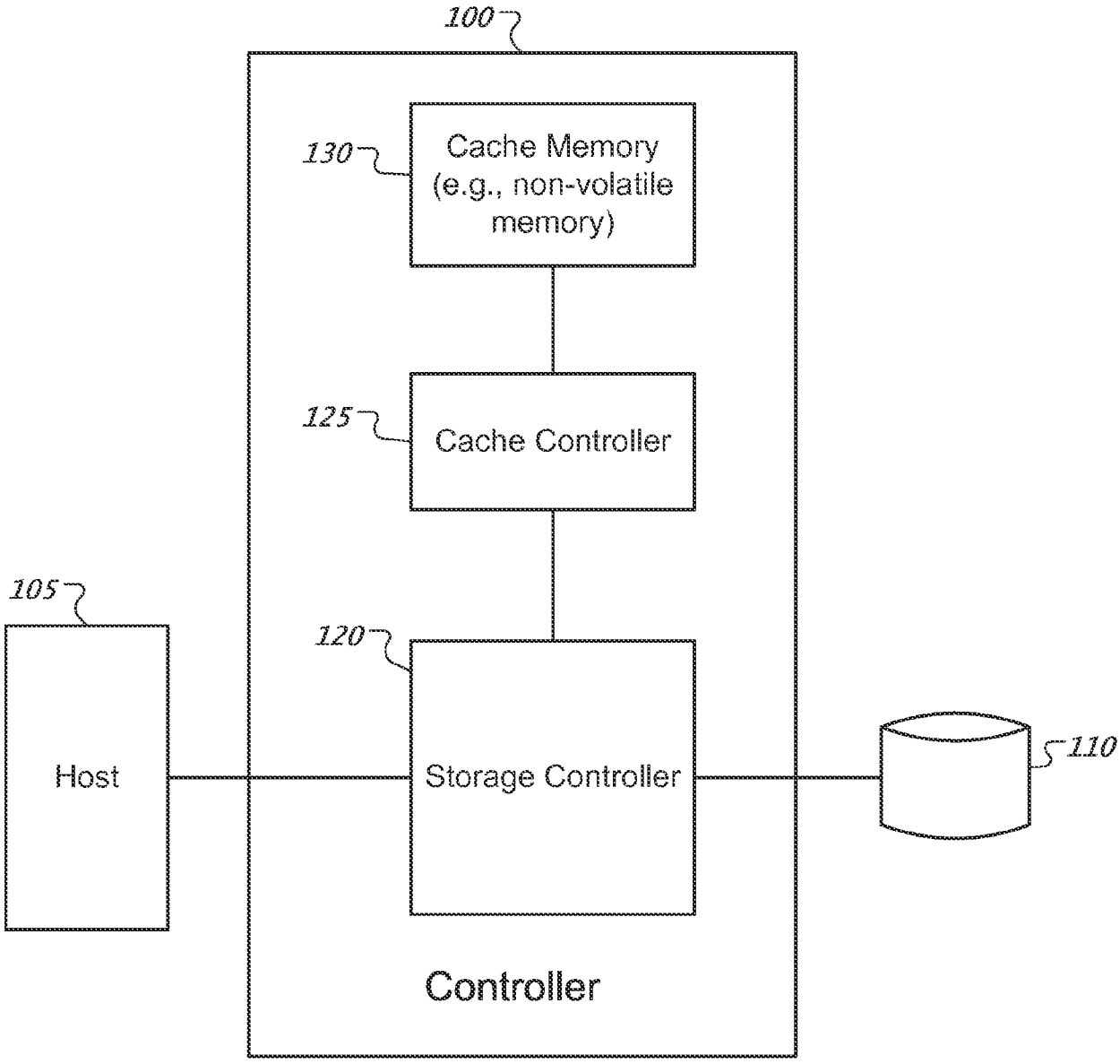

100
130
Cache Memory
(e.g., non-volatile
memory)
125
Cache Controller
105
120
Host
Storage Controller
110
Controller

FIG. 1

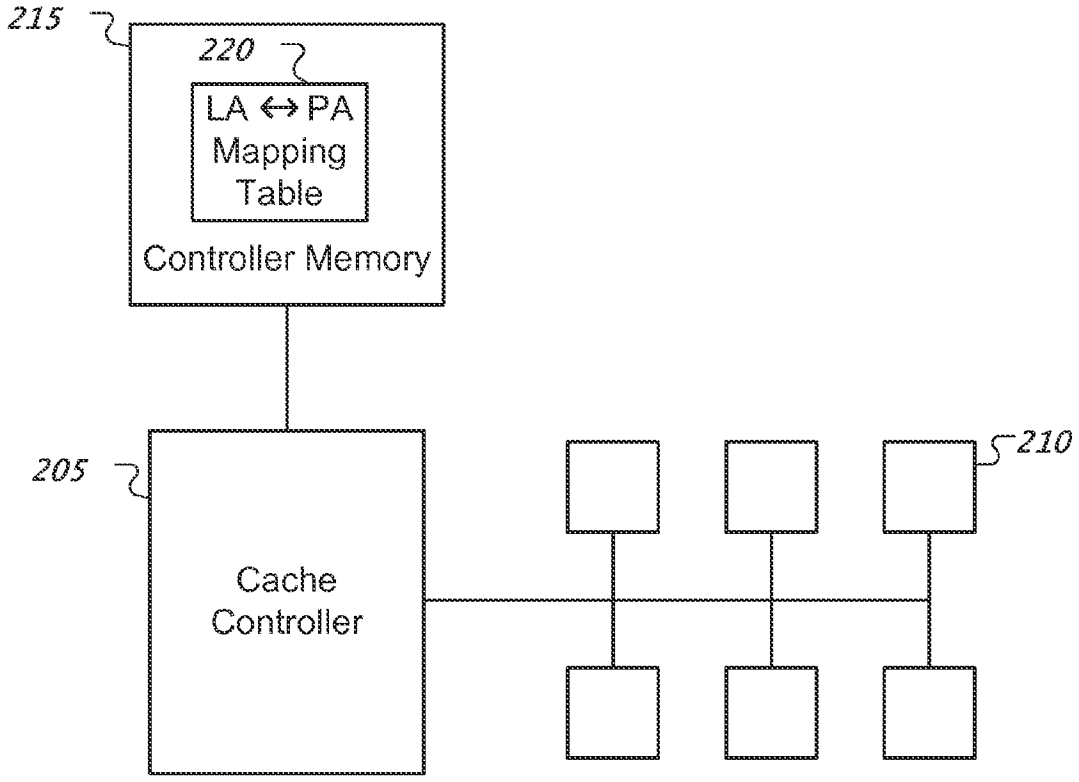

215
220
LA ↔ PA
Mapping
Table
Controller Memory
205
Cache
Controller
210

FIG. 2

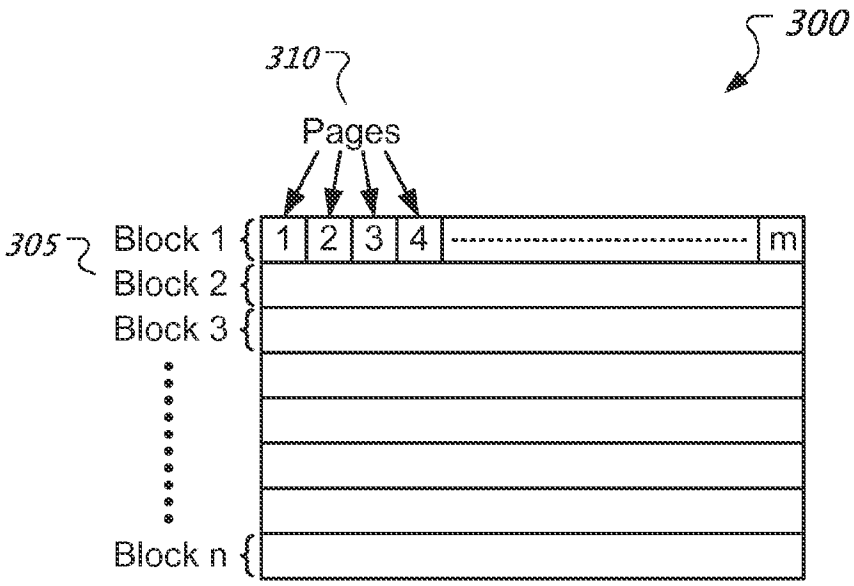


FIG. 3

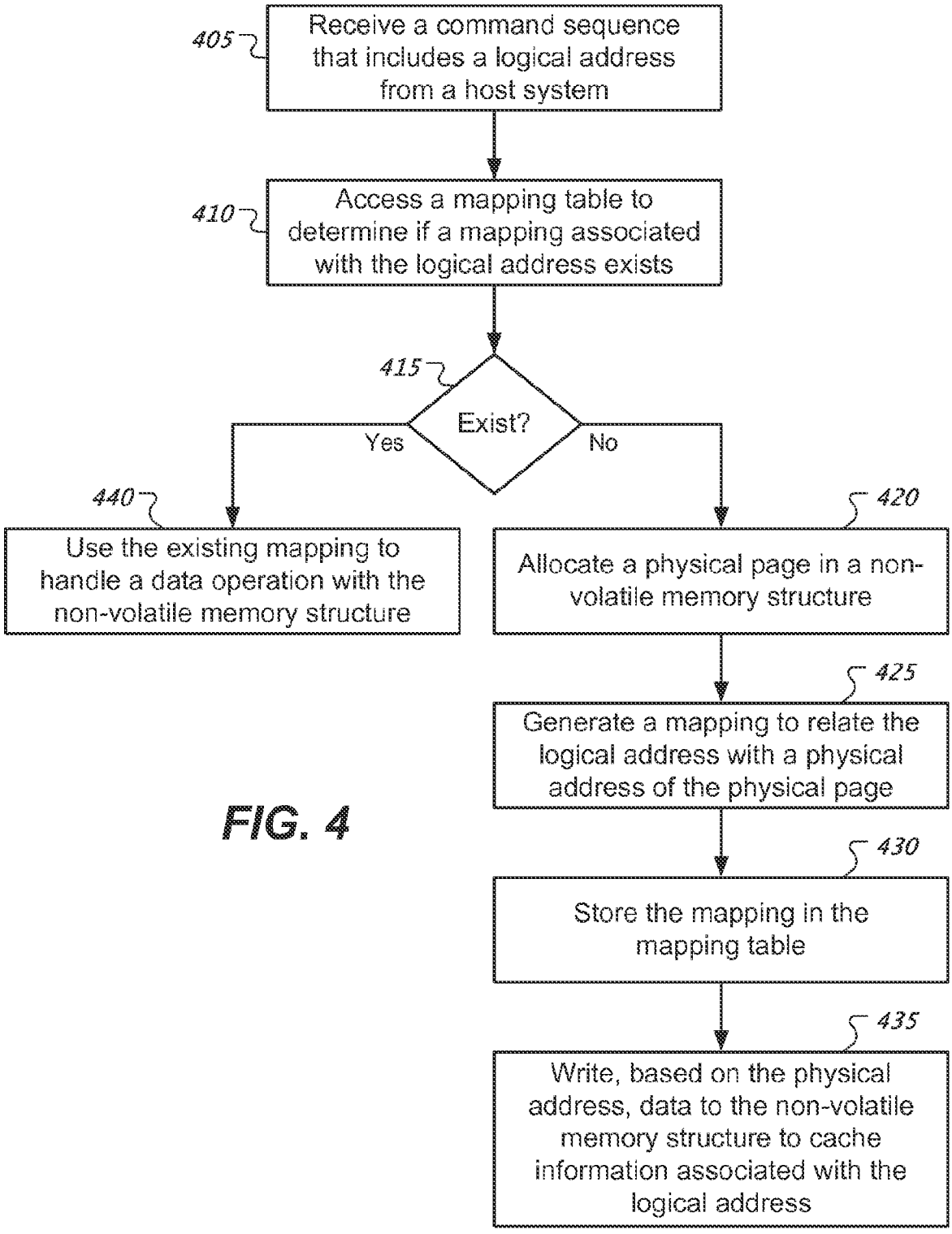

405
Receive a command sequence that includes a logical address from a host system
410
Access a mapping table to determine if a mapping associated with the logical address exists
415
Exist?
Yes
No
440
Use the existing mapping to handle a data operation with the non-volatile memory structure
420
Allocate a physical page in a non-volatile memory structure
425
Generate a mapping to relate the logical address with a physical address of the physical page
430
Store the mapping in the mapping table
435
Write, based on the physical address, data to the non-volatile memory structure to cache information associated with the logical address

FIG. 4

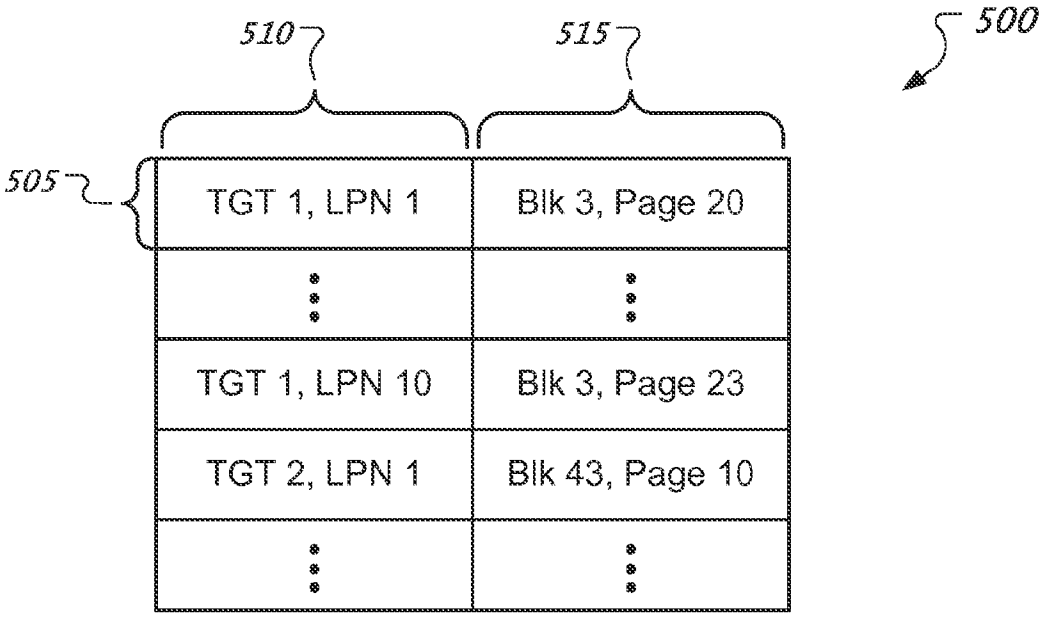
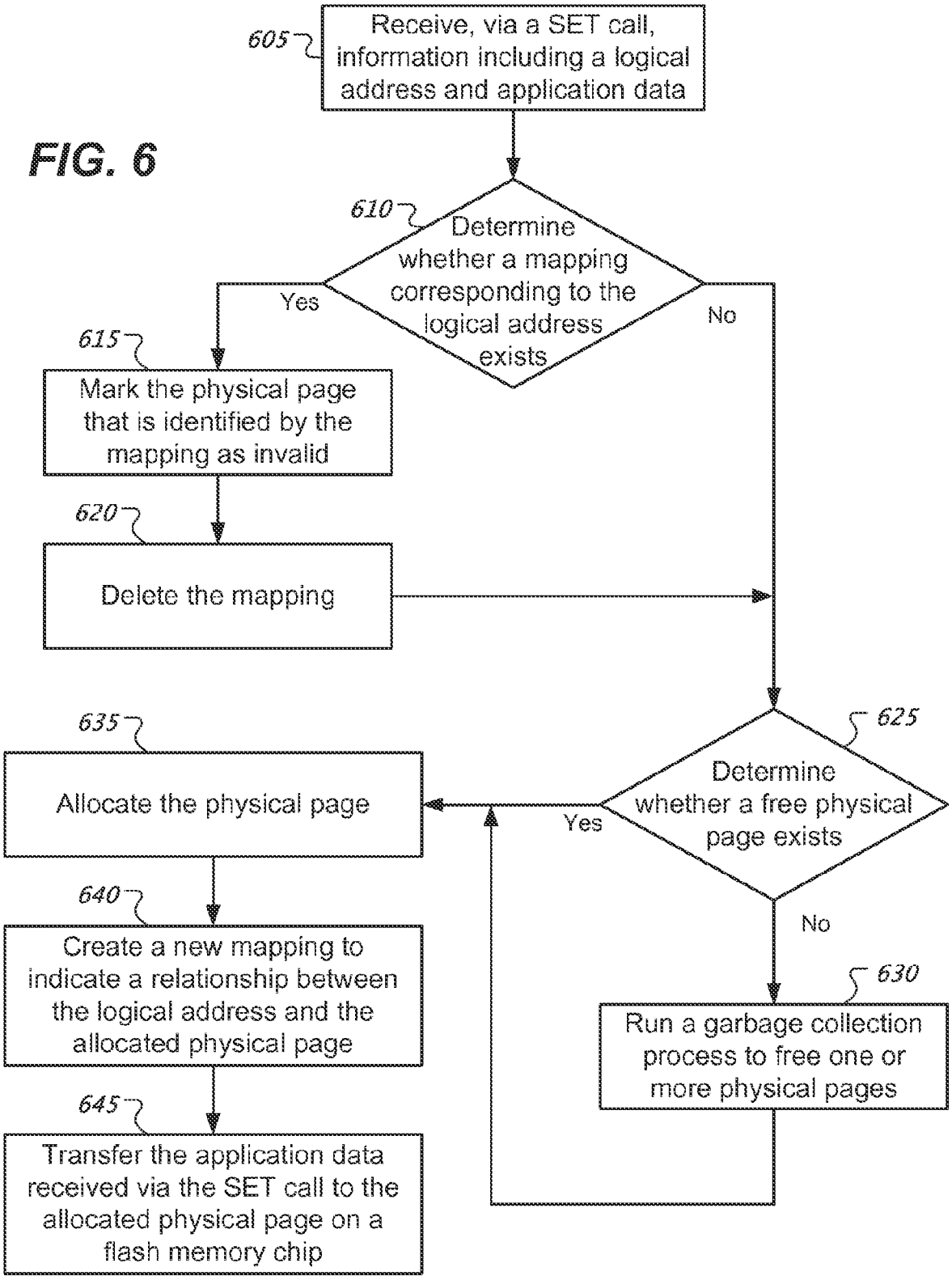

500
510
515
505
TGT 1, LPN 1
Blk 3, Page 20
TGT 1, LPN 10
Blk 3, Page 23
TGT 2, LPN 1
Blk 43, Page 10

FIG. 5

FIG. 6


605
Receive, via a SET call, information including a logical address and application data
610
Determine whether a mapping corresponding to the logical address exists
Yes
No
615
Mark the physical page that is identified by the mapping as invalid
620
Delete the mapping
625
Determine whether a free physical page exists
Yes
No
630
Run a garbage collection process to free one or more physical pages
635
Allocate the physical page
640
Create a new mapping to indicate a relationship between the logical address and the allocated physical page
645
Transfer the application data received via the SET call to the allocated physical page on a flash memory chip

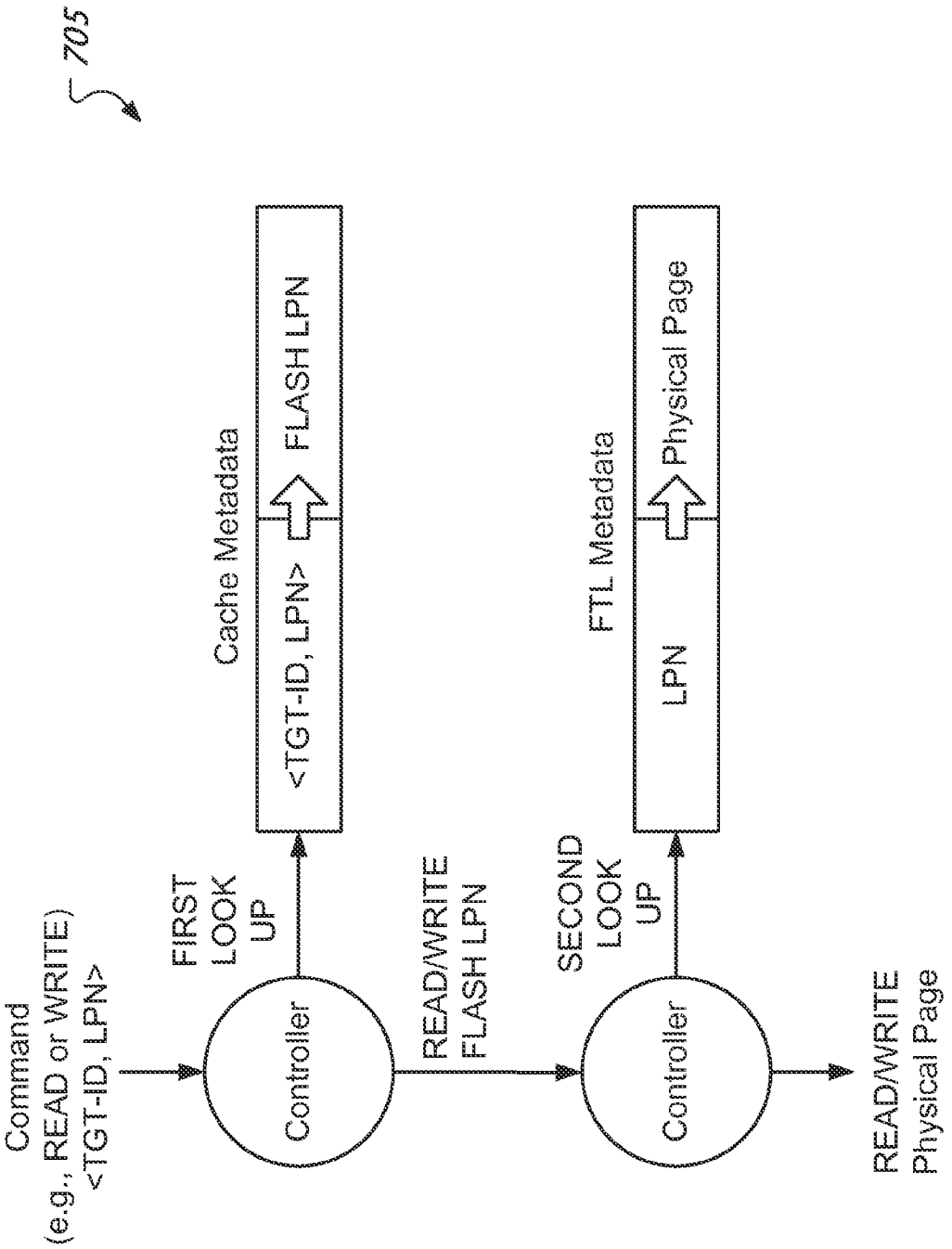

705
Command
(e.g., READ or WRITE)
<TGT-ID, LPN>
Controller
FIRST
LOOK
UP
Cache Metadata
<TGT-ID, LPN>
FLASH LPN
READ/WRITE
FLASH LPN
Controller
SECOND
LOOK
UP
FTL Metadata
LPN
Physical Page
READ/WRITE
Physical Page

FIG. 7A

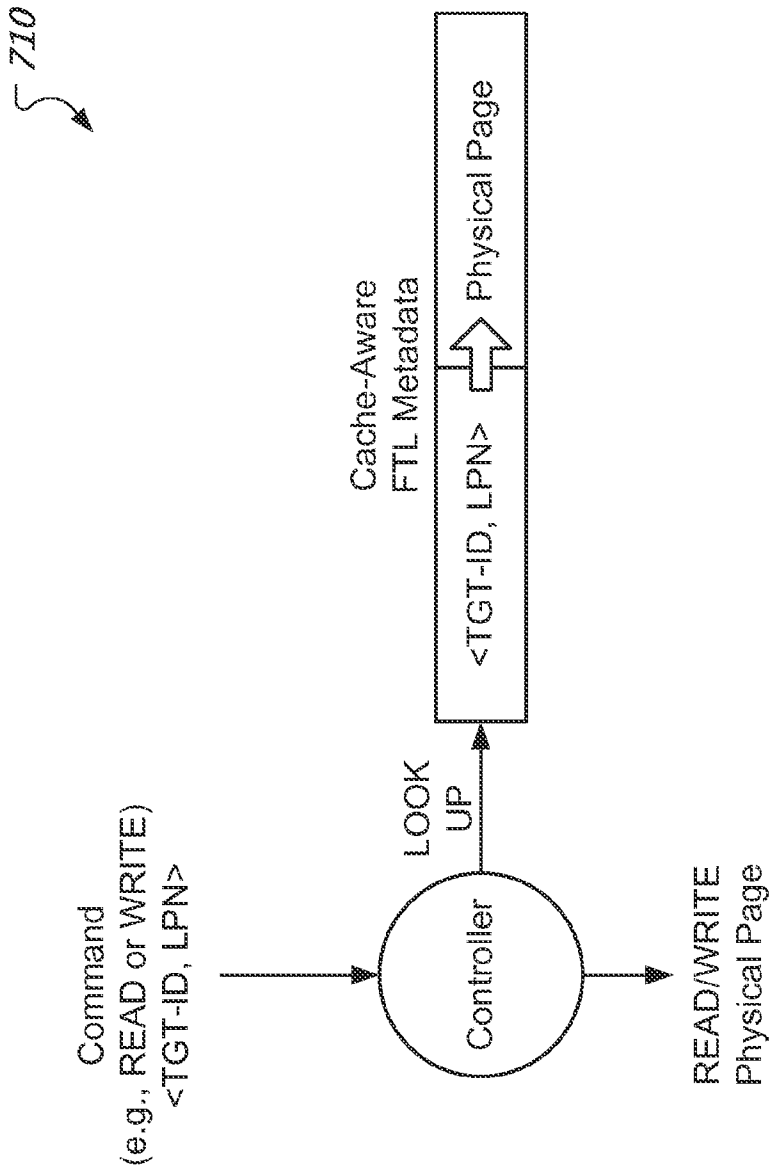

710
Command
(e.g., READ or WRITE)
<TGT-ID, LPN>
Controller
LOOK
UP
Cache-Aware
FTL Metadata
<TGT-ID, LPN>
Physical Page
READ/WRITE
Physical Page

FIG. 7B

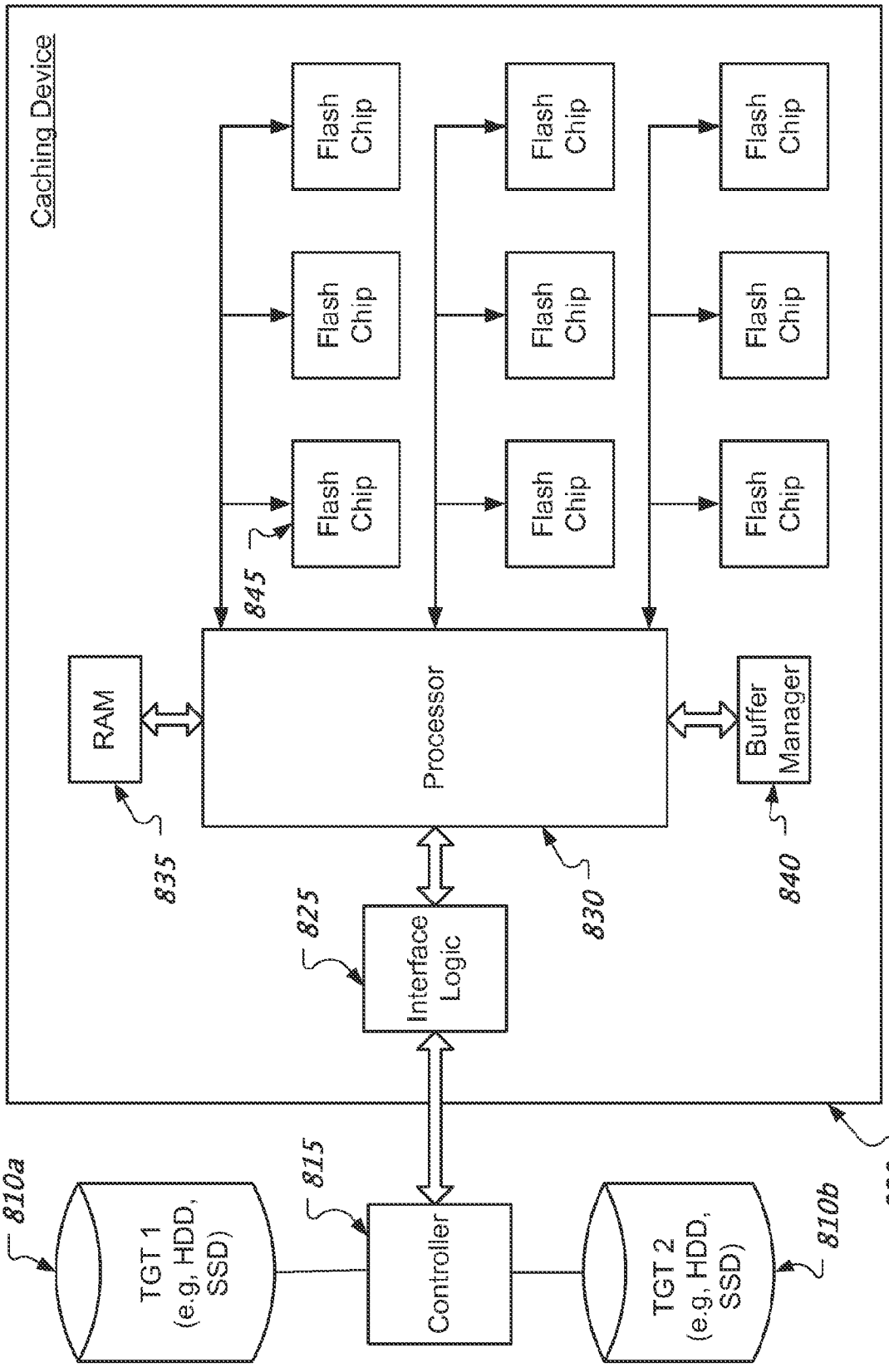

Caching Device
Flash Chip
Flash Chip
Flash Chip
Flash Chip
Flash Chip
Flash Chip
Flash Chip
Flash Chip
Flash Chip
845
RAM
Processor
Buffer Manager
835
830
840
825
Interface Logic
815
Controller
810a
TGT 1 (e.g, HDD, SSD)
TGT 2 (e.g, HDD, SSD)
810b
820

FIG. 8

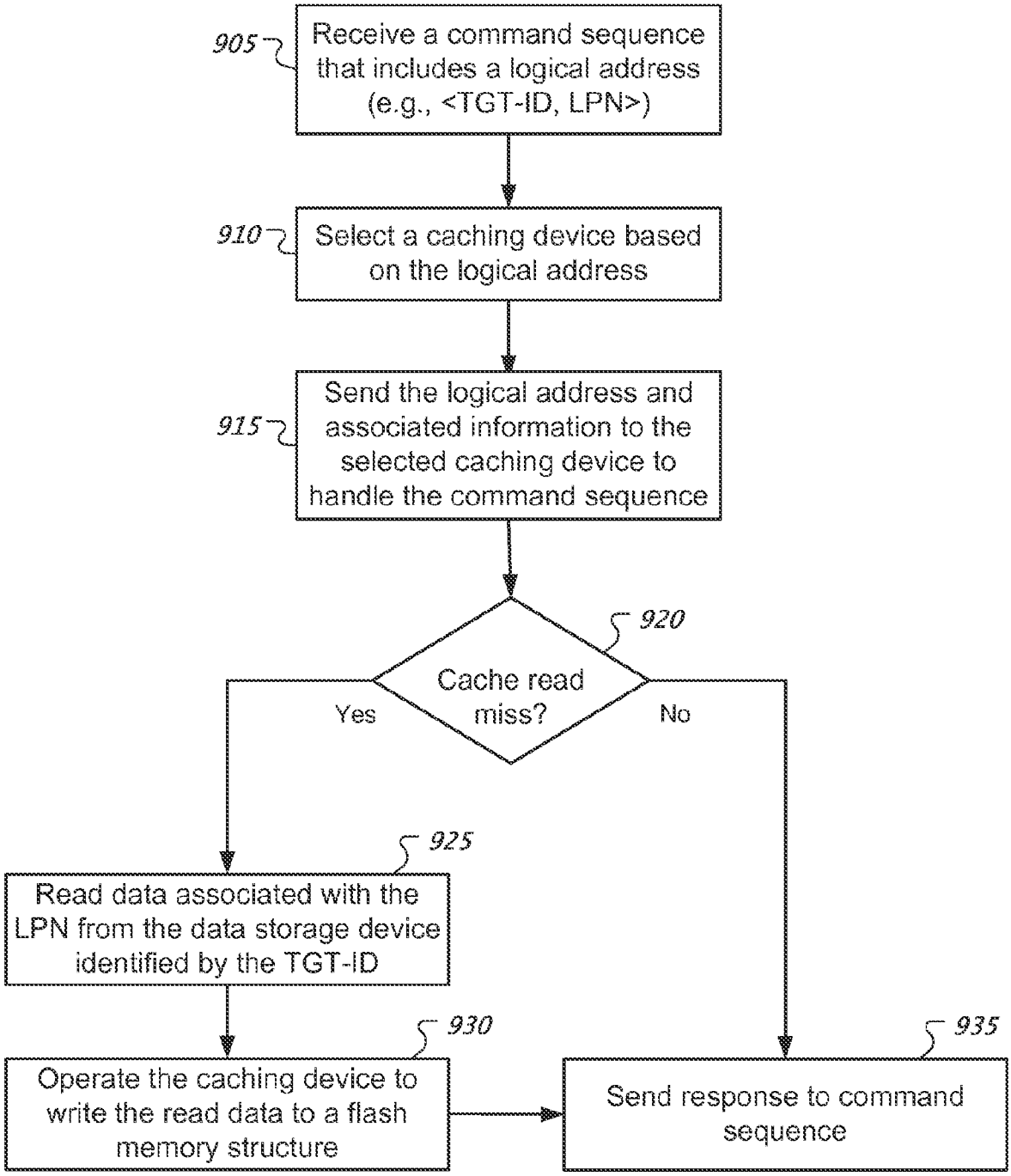

905
Receive a command sequence that includes a logical address (e.g., <TGT-ID, LPN>)
910
Select a caching device based on the logical address
915
Send the logical address and associated information to the selected caching device to handle the command sequence
920
Cache read miss?
Yes
No
925
Read data associated with the LPN from the data storage device identified by the TGT-ID
930
Operate the caching device to write the read data to a flash memory structure
935
Send response to command sequence

FIG. 9

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2010/051514

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F12/02 G06F12/08
ADD.

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

B. FIELDS SEARCHED

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

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

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

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 2007/146845 A2 (BITMICRO NETWORKS INC [US]; BRUCE REY H [US]; MATEO NOEME P [PH]; NITE) 21 December 2007 (2007-12-21) page 3, line 1 - line 17 page 13, line 1 - line 5 page 19, line 13 - line 19 page 21, line 12 - page 23, line 4 figure 2	1-20
X	EP 1 804 168 A1 (SAMSUNG ELECTRONICS CO LTD [KR]) 4 July 2007 (2007-07-04) paragraph [0009] paragraph [0018] - paragraph [0023] paragraph [0031]	1-4, 6-16, 18-20

-/--

[X] Further documents are listed in the continuation of Box C. [X] See patent family annex.

* Special categories of cited documents :

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

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

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
25 January 2011	07/02/2011

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Knutsson, Frédéric

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2010/051514

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2009/193182 A1 (NITTA TATSUO [JP]) 30 July 2009 (2009-07-30) paragraph [0022] - paragraph [0030] paragraph [0038] - paragraph [0040] -----	1-4, 6-16, 18-20

Form PCT/ISA/210 (continuation of second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2010/051514

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2007146845 A2	21-12-2007	NONE	
EP 1804168 A1	04-07-2007	CN 1991790 A	04-07-2007
		JP 2007179545 A	12-07-2007
		KR 20070068796 A	02-07-2007
		US 2007150654 A1	28-06-2007
US 2009193182 A1	30-07-2009	CN 101499036 A	05-08-2009
		JP 2009181314 A	13-08-2009

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
12 May 2011 (12.05.2011)

PCT

(10) International Publication Number
WO 2011/055407 A1

(51) International Patent Classification:
G06F 12/02 (2006.01)

(21) International Application Number:
PCT/JP2009/005859

(22) International Filing Date:
4 November 2009 (04.11.2009)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant *(for all designated States except US)*: **HITACHI, LTD.** [JP/JP]; 6-6, Marunouchi 1-chome, Chiyoda-ku, Tokyo, 1008280 (JP).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **TAKADA, Masanori** [JP/JP]; c/o Hitachi, Ltd., Systems Development Laboratory, 1099, Ohzenji, Asao-ku, Kawasaki-shi, Kanagawa, 2150013 (JP). **KITAHARA, Jun** [JP/JP]; c/o Hitachi, Ltd., Systems Development Laboratory, 1099, Ohzenji, Asao-ku, Kawasaki-shi, Kanagawa, 2150013 (JP).

(74) Agent: **WILLFORT INTERNATIONAL**; Kanda Center Bldg. 5F, 3-2, Kajicho 2-chome, Chiyoda-ku, Tokyo, 1010044 (JP).

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

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: FLASH MEMORY MODULE

(57) Abstract: Logical/physical conversion information is configured from first conversion information and second conversion information. A controller of a flash memory module restores the first conversion information at boot up time, enables an access command to be received from a host after having restored the first conversion information, and restores the second conversion information after an access command is able to be received.

[Fig. 6]

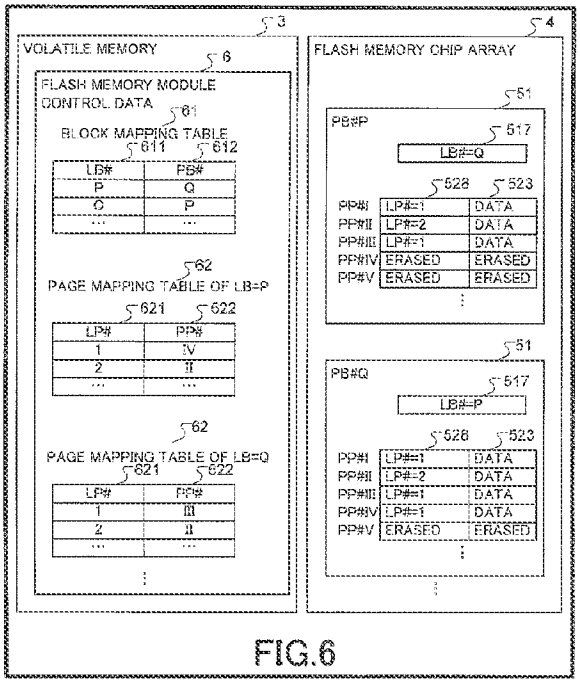


FIG.6

WO 2011/055407 A1

Published:

---- *with international search report (Art. 21(3))*

Description

Title of Invention: FLASH MEMORY MODULE

Technical Field

[0001] The present invention generally relates to a module comprising a flash memory.

Background Art

[0002] Lower-cost, higher-capacity flash memory is coming into widespread use. Thanks to outstanding performance and power consumption features, flash memory use is spreading especially rapidly in modules that place emphasis on these features (for example, storage systems and compact mobile modules).

[0003] A flash memory comprises a plurality of flash memory chips (hereinafter, chips), and each chip comprises a plurality of physical blocks. A physical block is comprised from a plurality of physical pages. The physical block and the physical page are both physical storage areas. For example, the reading and writing of data are carried out in page units, and the erasing of data is carried out in block units.

[0004] The flash memory, in particular, the large-capacity NAND-type flash memory has the following two characteristic features.

[0005] The first is the fact that it is not possible to write new data over data that has already been written to a physical page, and in order to write new data to this physical page, it is necessary to carry out an erase process for the physical block comprising this physical page. Data of a fixed size is written to the erased physical block in order from the start of this block. As described above, the flash memory is limited as to the size and order of a read and a write.

[0006] The other is a wear-out phenomenon. The wear-out phenomenon is one in which the flash memory becomes unable to hold information after erase processing has been carried out more that a certain number of times. The number of erasures is more apt to reach the limit in a case where writes are carried out frequently. In a case where writes concentrate on a specific area, the physical blocks comprising this area reach the end of their service life early on.

[0007] With respect to these problems, for example, Patent Literature 1 discloses a technology in which a controller for controlling the flash memory conceals the size and order limitations of reads and writes. In accordance with Patent Literature 1, an area for appending data is disposed inside the flash memory. Also, for example, Patent Literature 2 discloses a technology for extending the service life of the flash memory by dispersing write destinations.

[0008] These technologies require information (logical/physical conversion information) for making a logical address used by a host to specify an access destination correspond to

a storage-destination flash memory address (physical address). Failure to make this logical/physical conversion data nonvolatile the same as the data being written/read to/from the host will make it impossible to store the data correctly. For example, Patent Literature 3 discloses a technology for storing this logical/physical conversion information together with a user data, and restoring the original logical/physical conversion information from the stored data.

Citation List

Patent Literature

[0009] PTL 1: Japanese Patent Application Laid-open No. 2009-064251
PTL 2: Japanese Patent Application Laid-open No. 2007-265265
PTL 3: Japanese Patent Application Laid-open No. H8-077074

Summary of Invention

Technical Problem

[0010] Ordinarily, a flash memory module boots up when the power is turned ON, and restores the logical/physical conversion information at boot up time. A management information unit is required to restore the logical/physical conversion information. The management information unit, for example, is an information unit that is written into the overhead area of the physical page, and denotes which physical block comprises this physical page, and the logical page inside the logical block to which this physical page corresponds.

[0011] Although the logical/physical conversion information is able to be restored in accordance with the above-mentioned technology, all the management information units must be read in order to perform restoration. For this reason, the restoration of the logical/physical conversion information takes a long time. For example, in a case where the flash memory has a capacity of 256 GB (gigabytes) and the size of the physical page is 4 kB (kilobytes), each management information unit from 64 M (mega) (256 GB divided by 4 kB) of physical pages must be read out. Since the time required for readout is normally 100 microseconds per physical page, the restoration of the logical/physical conversion information requires around 6,400 seconds. This is an extremely long time compared to the time required to boot up a hard disk (for example, around 10 seconds). As flash memory capacity increases, the time required for booting up will most likely increase proportionally.

[0012] An object of the present invention is to reduce the time required to boot up a flash memory module.

Solution to Problem

[0013] The logical/physical conversion information comprises a first conversion information and a second conversion information. A flash memory module controller restores the

first conversion information at boot up time, making the flash memory ready to receive an access command from the host after the first conversion information has been restored, and after the flash memory is ready to receive an access command, the controller restores the second conversion information.

Advantageous Effects of Invention

[0014] The present invention makes it possible to reduce the time required to boot up the flash memory module.

Brief Description of Drawings

[0015] [fig.1]Fig. 1 shows the configuration of a flash memory module related to a first embodiment of the present invention.

[fig.2]Fig. 2 shows the internal configuration of a chip 5.

[fig.3A]Fig. 3A shows the status of a physical block #p in the initialized state.

[fig.3B]Fig. 3B shows an example of the state subsequent to new data being written to the physical block #p.

[fig.3C]Fig. 3C shows the state of the physical block #p subsequent to a merge operation being performed.

[fig.4]Fig. 4 shows the configuration of a flash memory module control data 6.

[fig.5]Fig. 5 shows the configuration of a module control function 7.

[fig.6]Fig. 6 shows an example of the corresponding relationship between tables 61 and 62 inside a volatile memory 3 and the state of a physical block inside a chip array 4.

[fig.7]Fig. 7 shows an example of the configuration of a block mapping table 61.

[fig.8]Fig. 8 shows an example of the configuration of a page mapping table 62.

[fig.9]Fig. 9 shows an example of a scheme for storing data in a physical block 51.

[fig.10]Fig. 10 shows a detailed example of the configuration of a physical page 52.

[fig.11]Fig. 11 shows an example of the configuration of a block metadata 515.

[fig.12]Fig. 12 shows an example of the configuration of a page metadata 526.

[fig.13]Fig. 13 shows an example of the flow of processing carried out by a host read service function 71.

[fig.14]Fig. 14 shows an example of the flow of processing carried out by a host write service function 72.

[fig.15]Fig. 15 shows an example of the flow of processing carried out by a page mapping table reconstruct function 74.

[fig.16]Fig. 16 shows an example of the flow of processing carried out by a power ON function 75.

[fig.17]Fig. 17 shows an example of the flow of processing carried out by an initialize function 76.

[fig.18]Fig. 18 shows an example of an exchange between a controller 2 and the chip array 4 in S726 of the host write service function 72.

[fig.19]Fig. 19 shows an example of the exchanges between the controller 2 and the chip array 4 in S728 through S730 of the host write service function.

[fig.20]Fig. 20 shows an example of the exchanges between the controller 2 and the chip array 4 in S752 of the power ON function 75 and in S743 of the page mapping table reconstruct function 74.

[fig.21]Fig. 21 shows an example of the corresponding relationship between a logical address and a logical block/logical page in the first embodiment of the present invention.

Description of Embodiments

[0016] A flash memory module in which a flash memory module related to the first embodiment of the present invention is used will be explained below by referring to the attached drawings. However, it should be noted that this embodiment is merely an example for realizing the present invention, and does not limit the technical scope of the present invention. Also, the same reference signs will be assigned to common configurations in the respective drawings.

[0017] Fig. 1 shows the configuration of a flash memory module related to the first embodiment of the present invention.

[0018] The flash memory module 1 comprises a flash memory controller (hereinafter, controller) 2, a volatile memory 3, a chip array (one or more flash memories) 4, a flash memory bus 8, and a host interface 9. The flash memory module 1, for example, may be the same shape as a 3.5-inch or 2.5-inch hard disk drive, or may be larger or smaller than same.

[0019] The chip array 4 comprises a plurality of flash memory chips (hereinafter, chip) 5.

[0020] The controller 2 controls the chip array 4, and also controls communications with a host 11. For example, the controller 2 comprises a processor for executing a flash memory control software (a computer program), a memory controller for writing/reading data to/from the chip 5, and a host interface controller for exchanging data with the host 11. In Fig. 1, the controller 2 comprises a single LSI (Large Scale Integration), but, for example, the controller 2 may also be comprised of a plurality of IC (Integrated Circuit) chips, such as an externally connected processor. The flash memory module control function (hereinafter, module control function) 7 of the controller 2 will be explained below.

[0021] The volatile memory 3 is used for storing, from among the information handled by the controller 2, the information required for rapid access. Specifically, for example, the volatile memory 3 stores the above-mentioned flash memory control software, and

a flash memory module control data (hereinafter, module control data) 6 used by this software. A detailed explanation of the module control data 6 will be given below. The volatile memory 3, for example, is a DRAM (Dynamic Random Access Memory). Furthermore, another type of storage medium may be used instead of a volatile memory 3 as long as it is capable of high-speed access.

[0022] The chip 5 is used for storing information written/read to/from the host 11. Data that should be nonvolatile is also stored from among the information required for control. This will be explained in detail below.

[0023] The flash memory bus 8 is used for connecting the controller 2 and the chip 5. In Fig. 1, a plurality of chips 5 are connected to one flash memory bus 8, but another connection scheme may be used.

[0024] The host interface 9 is used to connect the flash memory module 1 to the host 11, and to carry out the exchange of information. For example, a SATA (Serial ATA), SAS (Serial Attached SCSI), FC (Fibre Channel) or other such storage interface may be used as the host interface 9, and a PCI-Express or other such I/O-type interface may also be used.

[0025] The flash memory module 1 may be provided in the storage system. In this case, the storage system comprises a plurality of flash memory modules 1. A RAID group is comprised of two or more memory modules 1. A logical volume is created based on the RAID group. In accordance with this, the host 11 may be a controller (hereinafter, storage controller) of the storage system, or may be a host computer that is connected to the storage system. In the case of the former, the storage controller is able to send an access-destination logical address that accords with an I/O command from the host computer to the flash memory module 1 that manages this logical address, and to access the physical block that corresponds to this logical address.

[0026] Further, the flash memory module 1 may be a SSD (Solid State Drive) or a portable flash memory module. In this case, the host 11 may be a computer.

[0027] Fig. 2 shows the internal configuration of a chip 5. Furthermore, in Fig. 2 and the subsequent drawings, the physical block is abbreviated as "PB", the physical page is abbreviated as "PP", the logical block is abbreviated as "LB" and the logical page is abbreviated as "LP".

[0028] The chip 5 is comprised of a plurality of physical blocks 51. The physical block 51 is comprised of a plurality of physical pages 52.

[0029] The chip 5 has three interfaces, read, write and erase. Specifically, these are as follows.

[0030] (Read) It is possible to read in data from either an entire physical page 52, or a portion of a physical page 52.

[0031] (Write) It is possible to write data to an empty physical page (a physical page, which

has been erased, and on which nothing is written) 52 in order from the start of the physical block 51. Specifically, for example, for physical block 51, in which data is only written on physical page #1, a write is possible to physical page #2, but it is not possible to overwrite the data in physical page #1. It is also not possible to skip physical page #2 and write data to physical page #3.

[0032] (Erase) It is possible to erase data in physical block 51 units. That is, it is not possible to erase only one part inside the physical block 51, and make that part an unwritten state (that is, make it an empty block).

[0033] Normally, the size of the physical block, the size of the physical page, and the size of the read/write unit from the host 11 will differ. For example, as shown in Fig. 21, the physical block is 256 kB, the physical page is 2 kB, and the size of the unit being written/read to/from the host 11 (the access size) is 512 B. That is, a logical block is allocated for each range of logical addresses capable of being specified from the host 11, and a physical block is allocated to the logical block. One or more physical blocks may be allocated to one logical block, but in this embodiment, it is supposed that one physical block is allocated to one logical block. The physical block is configured from a plurality of physical pages. A physical page inside a physical block allocated to a logical block is allocated to each logical page configuring the logical block. The controller 2 is able to specify the logical page of the access destination by performing a predetermined operation based on the logical address specified in the access command from the host 11, and is able to specify the physical page corresponding to the access-destination logical page based on the module control data 6.

[0034] For this reason, a mechanism for making up the difference between the size of the access from the host 11 and the size of the write unit/erase unit (page/block) of the chip 5 is needed for a write.

[0035] Figs. 3A through 3C show examples of the controller 2 operations for concealing the difference between the chip 5 page/block size and the size of the access from the host 11. Furthermore, in Figs. 3A through 3C, a physical block 51 having p as the physical block number (PB#) is given as an example, but the configuration is the same for another physical block 51 as well. The physical block 51 with PB# of p will be referred to as "physical block #p" below.

[0036] A plurality of physical pages 52 inside the physical block #p are divided into a master area and an append area. That is, there are physical pages 52 that belong to the master area and physical pages 52 that belong to the append area in the physical block #p. The data written to each physical page 52 is managed using a logical page number (LP#).

[0037] Fig. 3A shows the status of the physical block #p in the initialized state.

[0038] In the initialized state, the data from a logical page #1 (the logical page having 1 as

the LP#) to a logical page #a are respectively written to each physical page 52 from a physical page #1 (the physical page having 1 as the PP#) to a physical page #a. For the remaining physical pages 52 (that is, from physical page #(a + 1) through physical page #n), the state (erased state) is one in which data has yet to be written in.

[0039] Fig. 3B shows an example of the state subsequent to new data having been written to the physical block #p.

[0040] In this example, the data to be written to logical page #a is stored in the physical page #(a + 1), and the data to be written to logical page #2 is stored in physical page #n. In so doing, it becomes possible to append data to the physical block #p without erasing the physical block #p (strictly speaking, without performing an erase process for the physical block #p) with respect to a write unit request that is smaller than the size of the physical block #p.

[0041] However, according to the example of Fig. 3B, since data is written up to the last physical page #n in the append area, a physical page that is capable of being written to does not exist in the physical block #p. Accordingly, the controller 2 carries out a merge operation (reclamation) that reflects the appended data in the master area. In the merge operation, the controller 2 reads in the latest data of each logical page from the physical block #p, and after carrying out an erase process for the physical block #p, writes the latest data for the logical page corresponding to this physical page in order from the first physical page of the master area. Furthermore, so that the data write is carried out in order from the first physical page to the last physical page, the latest data corresponding to the same logical page exists in either the last physical page or the physical page closest to the last physical page.

[0042] Fig. 3C shows the state of the physical block #p after the merge operation has been carried out.

[0043] The data is written in order from the first physical page of the master area, and all the physical pages comprising the append area are in an erased state. The first physical page #1 through the last physical page #a of the master area correspond to the logical pages #1 through #a. Therefore, for example, the data (the latest data) written in the physical page #1 prior to the merge operation is once again written to physical page #1. This is because the data of the logical page #1 has not been updated. The latest data of the logical page #2 (the data written to the append area) is written to the physical page #2.

[0044] Furthermore, the data write destination in the merge operation may be another empty block (an erased physical block) instead of the same physical block #p. Specifically, for example, the respective latest data corresponding to the respective logical pages inside the physical block #p may be copied to an empty block, and thereafter, an erase process may be carried out with respect to the physical block #p.

[0045] Fig. 4 shows the configuration of the module control data 6.

[0046] The module control data 6 is data used in controlling access to the chip array 4. The module control data 6 comprises a block mapping table 61 comprising information denoting the corresponding relationship between the logical blocks and the physical blocks, and a page mapping table 62 comprising information denoting the corresponding relationship between the logical pages and the physical pages. These tables 61 and 62 will be explained in detail below. Furthermore, the information of the tables 61 and 62 may be managed in a form other than a table.

[0047] Fig. 5 shows the configuration of the module control function 7.

[0048] The module control function 7, for example, is a software (a computer program) that is executed by the processor (a typical microprocessor) of the controller 2. The process carried out by the module control function 7 below is actually performed by the processor that executes this software. At least one part of the module control function 7 may be realized using a hardware circuit. The module control function 7 comprises the host read service function 71, the host write service function 72, the page mapping table reconstruct function 74, the power ON function 75, and the initialize function 76. The functions 71 through 76 will be explained in detail below.

[0049] Fig. 6 shows an example of the corresponding relationship between the tables 61 and 62 inside the volatile memory 3 and the state of the physical block inside the chip array 4.

[0050] In this embodiment, the logical block number and the logical page number are specified based on the logical address specified by the host 11, and access is carried out to the physical page corresponding to the specified logical block number and logical page number. Specifically, for example, in a case where an access from the host 11 to the flash memory module 1 is carried out using a LBA (Logical Block Address) in SCSI, the quotient obtained by dividing the LBA by the capacity of the master area inside the block is the logical block number, and the quotient obtained by dividing the remainder by the page capacity is the logical page number.

[0051] The block mapping table 61 comprises information denoting the corresponding relationship between each logical block number 611 and each physical block number 612. By referring to this table 61, it is possible to determine the physical block 51 in which the access-destination area corresponding to the logical address specified by the host 11 exists.

[0052] The page mapping table 62, for example, exists in each logical block. The page mapping table 62 comprises information denoting the corresponding relationship between each logical page number 621 and each physical page number 622. By referring to this table 62, it is possible to determine the physical page in which the access-destination area corresponding to the logical address specified by the host 11

exists.

[0053] The physical block 51 stores the logical block number 517 of the logical block to which this physical block 51 is allocated. The physical page 52 (for example, this user area) stores data (so-called user data) 523 that is written/read directly to/from the host 11. This physical page 52 (for example, this overhead area) also stores the logical page number 528 corresponding to this physical page 52.

[0054] Specifically, for example, in the overhead area of each physical page 52 there is stored metadata comprising the logical page number 528 corresponding to this physical page 52. As used in this embodiment, "metadata" is not the data written/read directly to/from the host 11, but rather refers to the data required for management purposes. Metadata, for example, is the management information unit.

[0055] For example, in addition to the logical page number 528 corresponding to the first physical page #i, the metadata stored in the overhead area of the first physical page #i may also include the logical block number 517 of the logical block to which is allocated the physical block 51 having this first physical page #i.

[0056] Fig. 7 shows an example of the configuration of the block mapping table 61.

[0057] This table 61 comprises information denoting the corresponding relationship (the block corresponding relationship) between the logical block numbers 611 and the physical block numbers 612. This table 61 is created when the power to the flash memory module 1 is turned ON, and is referred to at the time of a write/read to/from the host 11. This table 61 is updated in a case where the block corresponding relationship has changed. The block corresponding relationship, for example, changes in a case where wear-leveling for suppressing the bias of the number of erasures is carried out. The block corresponding relationship may also change in a case where a merge operation is performed for any physical block 51, and in a case where the page mapping table 62 has been reconstructed.

[0058] For each logical block number 611, the block mapping table 61 comprises a corresponding physical block number 612, and a page mapping table flag 613 showing whether or not a page mapping table 62 is available. In a case where a valid page mapping table 62 is available for a logical block, the value of the flag 613 corresponding to this logical block number 611 is "Y", and in a case where a valid page mapping table 62 is not available for a logical block, the value of the flag 613 is "N". Furthermore, the value of the flag 613, for example, is changed from "Y" to "N" in a case where the physical block allocated to the corresponding logical block has changed to a different physical block (that is, in a case where the merge operation has been carried out). For this reason, in a case where this logical block is to be accessed for the first time since this different physical block has been allocated to the corresponding logical block, the page mapping table 62 for this logical block is reconstructed.

[0059] Fig. 8 shows an example of the configuration of the page mapping table 62.

[0060] This table 62 comprises information denoting the corresponding relationship (page corresponding relationship) between the logical page numbers 621 and the physical page numbers 622. The above-mentioned block mapping table 61 is created when the power to the flash memory module 1 is turned ON, but this table 62 is not created when the power to the flash memory module 1 is turned ON, but rather is created in a case where a logical block specified based on a logical address is the access destination for the first time subsequent to the power to the flash memory module 1 having been turned ON. The table 62 is referred to at the time of a data write/read to/from the host 11, and is updated at the time of a merge operation or a data write from the host 11. The table 62 comprises a corresponding physical page number 612 for each logical page number 611. The table 62 also comprises the number (last appended physical page number) 623 of the physical page to which data was last appended of the plurality of physical pages comprising the append area.

[0061] Fig. 9 shows an example of a scheme for storing data in the physical block 51.

[0062] As described above, the plurality of physical pages 52 included in the physical block 51 are physical pages 52 belonging to the master area and physical pages 52 belonging to the append area. Each physical page 52 comprises an area (overhead area) in which metadata 521 is stored, and a user area, and the user area comprises a plurality of data areas (for example, sectors) 523 in which are stored a plurality of data 523.

[0063] A set of metadata inside the master area, that is, a set of metadata 521 of the physical page 52 belonging to the master area will be called "block metadata" 515 below. By contrast, the individual pieces of metadata 521 inside the append area, that is, the metadata 521 of the physical page 52 belonging to the append area will be called the "page metadata" 526. Incidentally, in a case where the total capacity of the metadata areas inside the master area is greater than the capacity of the block metadata 515 to be stored, at least one physical page 52 inside the master area may not be used to store the block metadata 515.

[0064] Fig. 10 shows a detailed example of the configuration of the physical page 52.

[0065] The physical page 52 comprises a metadata area in which the metadata 521 is stored, an ECC (Error Correcting Code) area in which the ECC 522 of this metadata 521 is stored, a plurality of data areas in which the plurality of data 523 are respectively stored, and an ECC area in which the ECC 524 of these respective data are stored.

[0066] As shown in Fig. 9, the metadata area stores a portion of the block metadata 515 in a case where the physical page 52 comprising this area belongs to the master area, and stores the page metadata 526 in a case where this physical page 52 belongs to the append area.

[0067] By providing the ECC 522 for the metadata 521 (specifically, the portion of the

block metadata 515 and the page metadata 526), it is possible to determine the validity (that is, whether or not an error has occurred) of the metadata 521 even in a case where only the metadata 521 has been read in.

[0068] Fig. 11 shows an example of the configuration of the block metadata 515.

[0069] The block metadata 515 comprises the number 517 of the logical block corresponding to the physical block 51.

[0070] Fig. 12 shows an example of the configuration of the page metadata 526.

[0071] The page metadata 526 comprises the number 528 of the logical page corresponding to the physical page 52 inside the append area. Since the number of the logical page corresponding to the physical page 52 inside the master area is uniquely stipulated, the corresponding logical page 528 need not be stored in the metadata area inside the master area.

[0072] Fig. 13 shows an example of the flow of processing carried out by the host read service function 71.

[0073] This function 71 is called when the controller 2 has received a read request from the host 11, specifies a location on the chip array 4 in which the specified data is stored, and transfers the data to the host 11. The details of this process will be explained on a step-by-step basis below.

[0074] Upon receiving a read request from the host 11, the function 71 determines the logical block number and the logical page number (S711). Specifically, for example, in a case where the logical address specified from the host 11 is an LBA or other linear address, the function 71 is able to let the quotient obtained by dividing the logical address by the logical block capacity be the logical block number, and is able to let the quotient obtained by dividing the remainder by the logical page capacity be the logical page number. The logical block number and the logical page number specified in S711 will be called the "read-source logical block number" and the "read-source logical page number" below.

[0075] Next, the function 71 refers to the block mapping table 61 on the volatile memory 3, and determines the physical block number (the read-source physical block number) 612 corresponding to the read-source logical block number 611 (S712).

[0076] Next, the function 71 checks whether or not a valid page mapping table 62 corresponding to the read-source logical block is available on the volatile memory 3 (S713). Specifically, for example, the function 71 refers to the block mapping table 61, and checks whether or not the flag 613 corresponding to the read-source logical block number is "Y". In a case where a valid page mapping table 62 is not available (S713: NO), the function 71 calls the reconstruct function 74. In accordance with this, a valid page mapping table 62 corresponding to the read-source logical block is constructed. Then, the function 71 proceeds to S715. Furthermore, in a case where a valid page

mapping table 62 is available in S713 (S713: YES), the function 71 proceeds to S715.

[0077] The function 71 refers to the page mapping table 62 corresponding to the read-source logical block, and determines the physical page number (the read-source physical page number) 622 corresponding to the read-source logical page number 621 (S715). In accordance with the above, the read-source physical block and the read-source physical page of the data requested by the host 11 are identified.

[0078] Next, the function 71 reads the data from the identified read-source physical page inside the identified read-source physical block (S716), and responds to the host 11 with the read data (S717).

[0079] In accordance with the above processing, in a case where a read request has been received from the host 11, first, the read-source physical block 51, which is the general storage location, is specified as an access destination, and thereafter, a page mapping table 62 denoting the corresponding relationship between the logical pages and the physical pages is constructed as necessary, and the read-source physical page 52 is specified. For this reason, the information denoting the detailed logical/physical corresponding relationship, that is, the table 62 denoting the corresponding relationship between the logical pages and the physical pages does not need to be created in advance. The block mapping table 61 and the page mapping table 62 are stored on volatile memory 3. The time required to access these tables is shorter than the time required to access the flash memory, thereby making it possible to hold read performance degradation in check.

[0080] Fig. 14 shows an example of the flow of processing carried out by the host write service function 72.

[0081] This function 72 is called when a write request has been received from the host 11, and stores the requested data in a suitable location. The details of this process will be explained on a step-by-step basis below.

[0082] Upon receiving a write request from the host 11, the function 72 determines the logical block number (the write-destination logical block number) and the logical page number (the write-destination logical page number) based on the logical address specified in the write request (S721).

[0083] Next, the function 72 refers to the block mapping table 61, and determines the physical block number (the write-destination physical block number) corresponding to the write-destination logical block number (S722).

[0084] Then, the function 72 refers to the flag 613 corresponding to the write-destination logical block number, and checks whether or not a valid page mapping table 62 corresponding to this logical block number is available (S723). In a case where this table 62 is not available (S723: NO), the function 72 calls the reconstruct function 74. In accordance with this, a page mapping table 62 corresponding to the write-destination

logical block is created. Then, the function 72 proceeds to S725.

[0085] The function 72 checks whether or not there are enough erased pages (empty physical pages) in the write-destination physical block (S725). Specifically, for example, the function 72 refers to the last appended physical page number 623 of the page mapping table 62 corresponding to the write-destination logical page. Then, the function 72 checks the number of empty physical pages inside the write-destination physical block by comparing this page number 623 against the maximum value of the physical page number (the number of the last physical page). The function 72 determines whether or not the amount of data (write data) that accords with the write request from the host 11 exceeds a capacity equivalent to the specified number of empty physical pages.

[0086] In a case where the result of the determination of S725 is affirmative, that is, in a case where there are enough erased pages (S725: YES), the function 72 writes the write data received from the host 11 to the page metadata 526 and to the erased pages (S726). The logical page number determined in S721 is included in the page metadata 526. The function 72, either in S732 or between 74 and S725, determines whether or not the data is stored in the physical page of the same number as the write-destination logical page number (that is, the corresponding physical page inside the master area), and in a case where the data is determined to be stored in this physical page, the function 72 may carry out the determination of S725. In a case where the data is determined not to be stored in this physical page, the function 72 is able to write the write data received from the host 11 to the physical page of the same number as the write-destination logical page number. In the case of a first write to the write-destination physical block, the function 72 may write the write data and the block metadata 515 to this physical block.

[0087] Subsequent to S726, the function 72 updates the page mapping table 62 corresponding to the write-destination logical block (S727). Specifically, for example, the function 72 updates the physical page number 622 of the entry (the entry inside the table 62) comprising the write-destination logical page number to the number of the write-destination physical page in S726, and, in addition, updates the last appended physical page number 623 to the number of the last write-destination physical page in S726. Then the function 72 proceeds to S734.

[0088] By contrast, in a case where there are not enough erased pages in S725 (S725: NO), the merge operation is performed. That is, the function 72 reads in the data (the latest data of the one or more pieces of data corresponding to the logical page) that exists in the write-destination physical block (hereinafter, the old physical block), merges this data with the write data received from the host 11, and creates a set of data to be written to the master area (S728). Next, the function 72 selects one unused physical

block (empty physical block) (S729). In this case, the function 72 may adopt a method of selecting a physical block which is not subjected to erasure so often in order to suppress the bias of the number of erasures among the physical blocks on the chip array 4. Then, the function 72 writes the data created in S728 to the selected physical block as well as to the block metadata 515 (S730). This block metadata 515 comprises the number of the logical block (write-destination logical block number) to which the old physical block is allocated. Then, the function 72 updates the page mapping table 62 corresponding to the write-destination logical block (S731). Specifically, for example, the function 72 performs updating such that the physical pages 622 corresponding to all the logical pages 621 point to the master area, and, in addition, updates the last appended physical page number 623 to the number of the first physical page of the append area. Next, the function 72 updates the block mapping table 61 (S732). Specifically, for example, the function 72 performs updating such that the physical block number 612 corresponding to the write-destination logical block number 611 points to the number of the new physical block written to in S730 (the physical block selected in S729). Next, the function 72 erases the data of the old physical block 51 (S733). Then, the function 72 proceeds to S734.

[0089] Lastly, the function 72 responds to the host 11 to the effect that the write has been completed (S734).

[0090] According to the above processing, in a case where a write request has been received from the host 11, the controller 2 is able to specify the write-destination physical block 51, which is the general storage location. Then, as necessary, the controller 2 is able to create a mapping table 62 denoting the corresponding relationship between the logical pages and the physical pages, and is able to store the data in a suitable location. In both a case where the corresponding relationship between the logical blocks and the physical blocks changes, and a case where the corresponding relationship between the logical pages and the physical pages changes, the controller 2, in addition to writing the write data to the flash memory, is also able to write as metadata information showing which is the write-destination logical block and which is the write-destination logical page. It is therefore possible to write the information for logical/physical conversion to the flash memory without generating excessive accesses. For this reason, it is possible to restore the block mapping table 61 and the page mapping table 62 from the metadata (the block metadata and the page metadata) written in the flash memory even when the volatile memory 3 has been erased due to a loss of power, and the block mapping table 61 and the page mapping table 62 have been erased. For example, restoration may be carried out only for the block mapping table 61, and thereafter, the page mapping table 62 may be restored as needed in accordance with a write access and a read access.

[0091] Fig. 15 shows an example of the flow of processing carried out in accordance with page mapping table reconstruct function 74.

[0092] This function 74 is called in a case where a page mapping table 62 corresponding to the logical block is not available on the volatile memory 3, and creates a page mapping table 62 based on the page metadata 526 on the flash memory. The details of the process will be explained on a step-by-step basis below.

[0093] Upon being called, the function 74 first performs initialization such that the logical page points to the physical page 52 of the master area in a page mapping table 62 corresponding to the called logical block (either the read-source logical block or the write-destination logical block) (S741). Specifically, for example, the function 74 performs the same processing for subsequent logical pages so that the first logical page points to the first physical page, and the second logical page points to the second physical page.

[0094] Next, the function 74 first reads in the page metadata 526 for all the physical pages 52 inside the append area (S742), and acquires the logical page number 528 inside this metadata 526 (S743). The function 74 also confirms that the read-in page metadata 526 is correct (confirms the presence or absence of an error in the metadata 526) by checking the ECC for the read-in page metadata 526 (S744). In a case where this metadata 526 is not correct, the ECC is used to correct this metadata 526. In a case where the data read-source physical page was the page that got erased, the function 74 proceeds to S747 since the page metadata has already been read from all the physical pages inside the append area. In a case where the data read-source physical page was not the page that got erased, appended data exists in this physical page, and the function 74 writes the number of this read-source physical page as the physical page number 622 corresponding to the logical page number specified in S743 for the entry (the entry inside the page mapping table 62) comprising the logical page number specified in S743 (S746). The function 74 repeats the above processing for the other physical pages 52 of the append area (S742), and proceeds to S747 when this processing is finished.

[0095] Lastly, upon finishing the updating of the physical page number 622 of the page mapping table 62, the function 74 updates the last appended physical page number 623 to the number of the last physical page to which the latest data has been written (S747).

[0096] According to the above steps, in a case where there is appended data, the physical page in which the latest data has been appended is associated with each logical page inside a certain specified logical block, and in a case where there is no appended data, the physical page of the master area is associated with each logical page inside the certain specified logical block. As result of this, it is possible to reconstruct (create) a page logical/physical corresponding relationship for one block. Since only the page metadata 526 of the append area may be read in this process, it is possible to minimize

the restoration overhead.

[0097] Furthermore, as has already been explained, the function 74 is called by the host read service function 71 and the host write service function 72 in the case of an access to a logical block for which a page mapping table 62 has not been created. However, when the controller 2 has excess throughput, the processing shown in Fig. 15 may be executed at a suitable time even without a request.

[0098] Fig. 16 shows an example of the flow of processing carried out by the power ON function 75.

[0099] This function 75 is called when the power to the flash memory module 1 is turned ON, and creates the block mapping table 61 on the basis of the metadata on the flash memory. The details of this process will be explained on a step-by-step basis below.

[0100] When the power is turned ON, first, the function 75 carries out the processing of S752 through S754 for all of the physical blocks 51. That is, the function 75 reads out the block metadata 515 from each physical block 51, and extracts the logical block number 517 from this metadata 515 (S752). Next, the function 75 checks to make sure this metadata 515 is correct by checking the ECC for the read-in block metadata 515 (S753). Then, in the block mapping table 61, the function 75 writes the read-source physical block number 612 of the metadata 515 into the entry corresponding to the logical block number 611 read in S752 (S754). The function 75 repeats the above processing for the other physical blocks 51 (S751), and proceeds to S755 when the processing is finished.

[0101] Next, the function 75 sets the page mapping table flags 613 of all the entries in the block mapping table 61 to "N", that is, to the value denoting unavailable (S755)

[0102] Lastly, the function 75 conveys to the host 11 the fact that the flash memory module 1 is ready to receive I/O (S756).

[0103] According to the above steps, the physical blocks corresponding to all the logical blocks inside the flash memory module 1 are identified. As a result, the block logical/physical corresponding relationship may be restored on the basis of information stored in the flash memory. Furthermore, in this process, since it is all right to read only the block metadata 515 stored in the master area, there is no need to read out the data received from the host 11 and the page metadata 526. It is therefore possible to minimize restoration overhead. As a result of this, it is possible to reduce the time required for boot up, thereby making it possible to convey to the host 11 in a short period of time that fact that I/O are ready to be received.

[0104] Fig. 17 shows an example of the flow of processing carried out by the initialize function 76.

[0105] This function 76 is used when the flash memory module 1 is used for the first time, and when erasing already stored data to return to the unused state once again. The

details of this process will be explained on a step-by-step basis below.

[0106] Upon being called, the function 76 first carries out an erase process for all the physical blocks 51 (S761).

[0107] The function 76 selects an unused physical block (an empty physical block) 51 for a single logical block from among the unprocessed logical blocks (S763).

[0108] Next, the function 76 creates the block metadata 515 and the ECC thereof (S764). The number of the above-mentioned single logical block being used in this process is stored in the block metadata 515.

[0109] Next, the function 76 stores the created block metadata 515 and data in the physical pages 52 of the master area (S765). The "data" referred to here may be a fixed value such as that utilized in an HDD format or the like.

[0110] Finally, the function 76 updates the physical block number 612 to the number of the physical block 51 selected in S763 for the block mapping table 61 entry corresponding to the logical block for which this process was carried out (S766).

[0111] The function 76 repeats the above-mentioned S763 through S766 for all the logical blocks (S762).

[0112] According to the above-mentioned process, corresponding physical blocks 51 are allocated to all the logical blocks, and the block metadata 515 comprising the numbers of the logical blocks corresponding to these physical blocks 51 are stored in the allocated physical blocks 51. As a result, it becomes possible to suitably process an access from the host 11 thereafter. It also becomes possible to properly restore the logical/physical conversion information from the flash memory 4 when the power is turned ON.

[0113] Fig. 18 shows an example of the exchange between the controller 2 and the chip array 4 in S726 of the host write service function 72.

[0114] The controller 2 sends the write command together with the data to be written to the physical page 52. The data to be written comprises the page metadata 526 and the ECC 522 thereof, and the data 523 sent from the host and the ECC 524 thereof.

[0115] Since the entire metadata write is carried out concomitantly with the write of the data sent from the host here, the metadata write does not substantially affect the write performance or number of writes-based service life of the flash memory.

[0116] Fig. 19 shows an example of the exchanges between the controller 2 and the chip array 4 in S728 through S730 of the host write service function.

[0117] In this drawing, the physical block a is the old physical block, and the physical block b is the physical block that is to be used anew.

[0118] First, the latest data is read out from the physical page 6 in which the latest data inside the physical block a is stored. This corresponds to the operation carried out in S728.

[0119] Then, a write command for the physical page 52 inside the physical block b (the physical page 52 inside the master area) is sent together with the data to be written to this physical page 52. The data comprises a portion of the block metadata 515 and the ECC 522 thereof, and the data 523 sent from the host 11 and the ECC 524 thereof. This corresponds to S730.

[0120] Since the entire metadata write is carried out concomitantly with the write of the data sent from the host 11 here, the metadata write does not substantially affect the write performance or number of writes-based service life of the flash memory.

[0121] Fig. 20 shows an example of the exchanges between the controller 2 and the chip array 4 in S743 for the page mapping table reconstruct function 74 and S752 for the power ON function 75.

[0122] When the power is turned ON, first the block metadata 515 is read out from the physical block 51. Specifically, the metadata 521 and the ECC 522 thereof are read out from the physical pages 52 of the master area. This process is repeated for all of the physical blocks 52. The preceding corresponds to S752 of the power ON function 75. Then, when the read and construction are finished, the host 11 is notified to the effect that the flash memory module 1 is ready to receive I/O.

[0123] Thereafter, upon receiving an access request (I/O request) from the host 11, the page metadata 526 is read out from the specified physical block 51 on the basis of the logical address specified in this access request. Specifically, the metadata 521 and the ECC 522 thereof are read out from the physical page 52 inside the append area. This process is repeated for the physical pages 52 inside the append area. The preceding corresponds to S743 of the page mapping table reconstruct function 74.

[0124] Only the block metadata 515, that is, only the metadata 521 inside the master area is read out here when the power is turned ON; the page metadata 526 is not read out. For this reason, the time required for boot up is reduced, thereby making it possible to notify the host 11 to the effect that the flash memory module 1 is ready to receive I/O in a short period of time.

[0125] Also, in a case where an access request is received from the host 11, and a page mapping table 62 corresponding to the logical block specified on the basis of the logical address specified in this access request is not available, only the page metadata 526 is read out and the table 62 is created. For this reason, it is possible to restore the information for the logical/physical conversion using a minimal read.

[0126] According to the embodiment described above, it is possible to reduce the affects on the write performance of the flash memory during normal operation while cutting down on the time required for boot up.

[0127] One embodiment of the present invention has been explained above, but it goes without saying that the present invention is not limited to this embodiment, and that

various changes are possible without departing from the gist thereof.

[0128] Furthermore, the page mapping table 62 may be prepared for each plurality of logical blocks instead of for each logical block. For example, when the controller 2 receives an access command, a page mapping table 62 corresponding to the access-destination logical block, which is specified on the basis of the logical address specified in this access command, and the next logical block may be created.

[0129] Also, a single master area and a single append area may be configured using two or more physical blocks. For example, a first physical block may be the master area, and a second physical block may be the append area.

[0130] The block mapping table 61 is constructed on the volatile memory 3, and a plurality of methods are conceivable as this construction method.

[0131] A first method stores a physical block number 612 in the volatile memory 3, and computes an address value on the volatile memory 3 in order to determine the physical block number 612 at access. This address value is determined from the logical block number 611 by adding a value obtained by multiplying the logical block number 611 by the number of bytes of configuration information in one entry of the block mapping table 61 to the start address value on the volatile memory 3 of the block mapping table 61. This address value is uniquely computed. In a case where either a read or write request is received from the host, it is possible to use this address value, access the volatile memory 3, and obtain the information of the physical block number 612 and the flag 613. This method is characterized in that it is possible to shorten the reference time.

[0132] A second method stores the information of the logical block number 611, the physical block number 612 and the flag 613 in the volatile memory 3 of the block mapping table 61 of the first method. This second method has the effect of enhancing reliability by making it possible to check that the address value computed from the logical block number 611 in the first method is correct by comparing this address value to the logical block number 611 stored in the volatile memory 3, and making it possible to detect a controller 2 logical error and a volatile memory 3 software error.

[0133] A third method stores the information of the logical block number 611, the physical block number 612 and the flag 613 in the volatile memory 3, and constructs a block mapping table 61 equivalent to the number of logical blocks being used. In a case where a logical block number 612 is not stored in a physical block, it is determined that the logical block is not being used. In this method, the effect is that the block mapping table 61 may be constructed on the capacity of the logical blocks being used, thereby making it possible to reduce the capacity of the block mapping table 61.

[0134] Also, as for S712 of Fig.13,there are a plurality of methods for determining the physical block number 612 in accordance with the method for constructing the block

mapping table 61. For example, there is a method for determining an address in the block mapping table 61 and finding the physical block number 612 that is stored there by carrying out a computation based on the read-source logical block number 611. In addition, there is also a method for storing the value of the read-source logical block number 611 in the block mapping table 61 as well, and checking to make sure that the computation is correct. Further, there is a method for storing a pair of the values of the read-source logical block number 611 and the physical block number 612 in the block mapping table 61, and using the read-source logical block number 611 to search inside the block mapping table 61 to find the corresponding physical block number 612.

[0135] Also, as for S722 of Fig.14, there are a plurality of methods for determining the physical block number 612 in accordance with the method for constructing the block mapping table 61. There is a method for determining the address in the block mapping table 61 and finding the physical block number that is stored there by carrying out a computation based on the write-destination logical block number, there is also a method for storing the value of the write-destination logical block number in the block mapping table 61 as well, and checking to make sure that the computation is correct, and furthermore, there is a method for storing a pair of the values of the write-destination logical block number and the physical block number in the block mapping table 61, and using the write-destination logical block number to search inside the block mapping table 61 to find the corresponding physical block number.

[0136] When constructing the block mapping table 61 in S754 of Fig.16, there are a plurality of methods for writing data into the block mapping table 61 in accordance with the method for constructing the block mapping table 61.

[0137] A first method determines an address in the block mapping table 61 by carrying out a computation based on a logical block number 517 selected from the metadata 515 and stores the physical block number 611 at this address.

[0138] A second method, in addition to the processing in the first method, also stores the logical block number 517 in the block mapping table 61 for checking the computed address.

[0139] A third method stores a pair comprising the logical block number 517 and the physical block number 611 in the order of the physical blocks 51 from which the block metadata 515 has been read out. In a case where a logical block number 517 is not stored in the physical block 51, that is, a case in which the physical block 51 is not being used, neither the logical block number 517 nor the physical block number 611 is stored in the block mapping table 61.

Reference Signs List

[0140] 1 Flash memory module

Claims

[Claim 1] A flash memory module, comprising:
a flash memory; and
a controller receiving an access command from a host, and, based on logical/physical conversion information denoting a corresponding relationship between a logical address and a physical address of the flash memory, accessing an area in accordance with a physical address corresponding to a logical address specified in the received access command,
wherein the logical/physical conversion information comprises first conversion information and second conversion information, and
wherein the controller restores the first conversion information at boot up time, enables an access command to be received after the first conversion information has been restored, and restores the second conversion information after an access command is able to be received.

[Claim 2] A flash memory module according to claim 1,
wherein the flash memory is a flash memory in which an access is carried out in page units and an erase is carried out in block units,
wherein the flash memory comprises a plurality of physical blocks,
wherein each physical block comprises a plurality of physical pages,
wherein the first conversion information is block mapping information denoting a corresponding relationship between logical blocks and physical blocks,
wherein the second conversion information is page mapping information denoting a corresponding relationship between logical pages and physical pages,
wherein the page mapping information is information restored for each of one or more physical blocks,
in a case where user data is to be written for the first time to an empty physical block, which is a physical block in an erased state, wherein the controller writes a block metadata, which is data comprising an ID of the logical block to which this empty physical block is allocated, to this empty physical block in addition to the user data,
in a case where a certain user data inside the physical block is to be updated, wherein the controller writes, in addition to a post-update user data, a page metadata, which is data comprising an ID of a write-destination logical page of the post-update user data, to a physical page

other than the physical page in which the certain user data is stored, wherein the controller restores the block mapping information at the boot up time by reading in the block metadata from each physical block of the flash memory, and

wherein the controller:

(W) in a case where a write command is received from the host after an access command is able to be received;

(w1) specifies a write-destination logical block and a write-destination logical page based on a logical address specified in the received write command;

(w2) specifies a physical block corresponding to the write-destination logical block based on the block mapping information;

(w3) determines whether or not valid page mapping information corresponding to the write-destination logical block is available;

(w4) in a case where the result of the determination of the (w3) is negative, creates the page mapping information corresponding to the write-destination logical block by reading in the page metadata from the physical page inside the physical block corresponding to the write-destination logical block;

(w5) stores a target user data of the received write command and the page metadata comprising an ID of the write-destination logical page either in the physical page inside the physical block corresponding to the write-destination logical block, or the physical page inside another physical block associated with the write-destination logical block, and updates the page mapping information corresponding to the write-destination logical block, and

(R) in a case where a read command is received from the host after an access command is able to be received;

(r1) specifies a read-source logical block and a read-source logical page based on a logical address specified in the received read command;

(r2) specifies a physical block corresponding to the read-source logical block based on the block mapping information;

(r3) determines whether or not a valid page mapping table corresponding to the read-source logical block is available;

(r4) in a case where the result of the determination of the (r3) is negative, creates the page mapping information corresponding to the read-source logical block by reading in the page metadata from the physical page inside the physical block corresponding to the read-

source logical block; and

(r5) specifies the physical page corresponding to the write-destination logical page based on the page mapping information corresponding to the write-destination logical block, and reads out the user data from the specified physical page.

[Claim 3] A flash memory module according to claim 2,

wherein when storing the block metadata, the controller stores a part of the block metadata that is stored in the physical page together with an ECC (Error Correcting Code),

wherein when storing the page metadata, the controller also stores the ECC in addition to the page metadata,

wherein when reading in the block metadata, the controller determines the presence or absence of an error in each part by using the ECC read in together with the each part of the block metadata, and

wherein when reading in the page metadata, the controller determines the presence or absence of an error in this page metadata by using the ECC read in together with this page metadata.

[Claim 4] A flash memory module according to claim 2,

wherein a master area and an append area are configured from in or more physical blocks, and

wherein the block metadata is stored in the master area, and the page metadata is stored in the append area.

[Claim 5] A flash memory module according to claim 4, wherein the controller, for each logical page, writes a user data, which is not the post-update user data, to the physical page belonging to the master area, and writes the post-update user data and the page metadata comprising an ID of the write-destination logical page of this post-update user data to the physical page belonging to the append area.

[Claim 6] A flash memory module according to claim 1,

wherein the flash memory is a flash memory in which an access is carried out in area units and an erase is carried out in area group units comprising a plurality of areas,

wherein the first conversion information is information denoting a corresponding relationship between logical areas and physical areas, and

wherein the second conversion information is information denoting a corresponding relationship between logical area groups and physical area groups.

[Claim 7] A flash memory module according to claim 6, wherein the second

conversion information is information restored for each of one or more area groups.

[Claim 8] A flash memory module according to claim 7,
wherein the area group is a block, and
wherein the controller receives an access command from the host, and in a case where the valid second conversion information is not available for the block specified based on the logical address specified in this access command, restores the second conversion information for one or more blocks comprising this block.

[Claim 9] A flash memory module according to claim 8,
wherein a master area and an append area are configured in one or more physical blocks, and
wherein the controller:
(X) when writing the user data to the append area, also writes information required to restore the second conversion information to the append area; and
(Y) when writing the user data to the master area, also writes information required to restore the first conversion information to the master area.

[Claim 10] A flash memory module according to claim 7,
wherein the controller stores, in respective one or more physical area groups, first management information comprising information denoting one or more logical area groups to which the one or more physical area groups are allocated,
wherein the controller stores, in a physical area in the physical area group, second management information comprising information denoting a logical area allocated to this physical area,
wherein the controller restores the first conversion information by reading in the first management information from the respective one or more physical area groups, and
wherein the controller restores, for one or more logical area groups, the second conversion information corresponding to the one or more logical area groups by reading in the second management information from each physical area in which the second management information is stored.

[Claim 11] A flash memory module according to claim 10,
wherein when storing the first management information, the controller stores a part of the first management information that is stored in the

physical area together with an ECC (Error Correcting Code),
wherein when storing the second management information, the controller also stores the ECC in addition to the second management information,
wherein when reading in the first management information, the controller determines the presence or absence of an error in each part by using the ECC read in together with the each part of the first management information, and
wherein when reading in the second management information, the controller determines the presence or absence of an error in this second management information by using the ECC read in together with this second management information.

[Claim 12] A flash memory module according to claim 1, wherein the controller restores the second conversion information in a case where the controller receives an access command from the host.

[Claim 13] A flash memory module according to claim 1, wherein the flash memory is configured from one or more physical blocks, the controller is connected to a memory, and the controller reads out a logical address from the physical block at boot up time, and stores the logical address and a physical address that corresponds to this logical address in the memory as the first conversion information.

[Claim 14] A flash memory module according to claim 13, wherein the controller reads out the logical address from the physical block at boot up time, and calculates a first address value corresponding to the logical address storage location from the logical address and a start address value of the memory,
the controller calculates a second address value from the start address value and the logical block address stored in the memory, and
the controller confirms that the first address value and the second address value match.

[Claim 15] A control method of a flash memory module receiving an access command from a host, and, based on logical/physical conversion information denoting a corresponding relationship between a logical address and a physical address of a flash memory, accessing an area in accordance with a physical address corresponding to a logical address specified in the received access command, the method comprising the steps of:
restoring first conversion information of the logical/physical conversion

information at boot up;
notifying the host that an access command is able to be received after having restored the first conversion information; and
restoring second conversion information of the logical/physical conversion information after having notified the host that an access command is able to be received.

[Fig. 1]

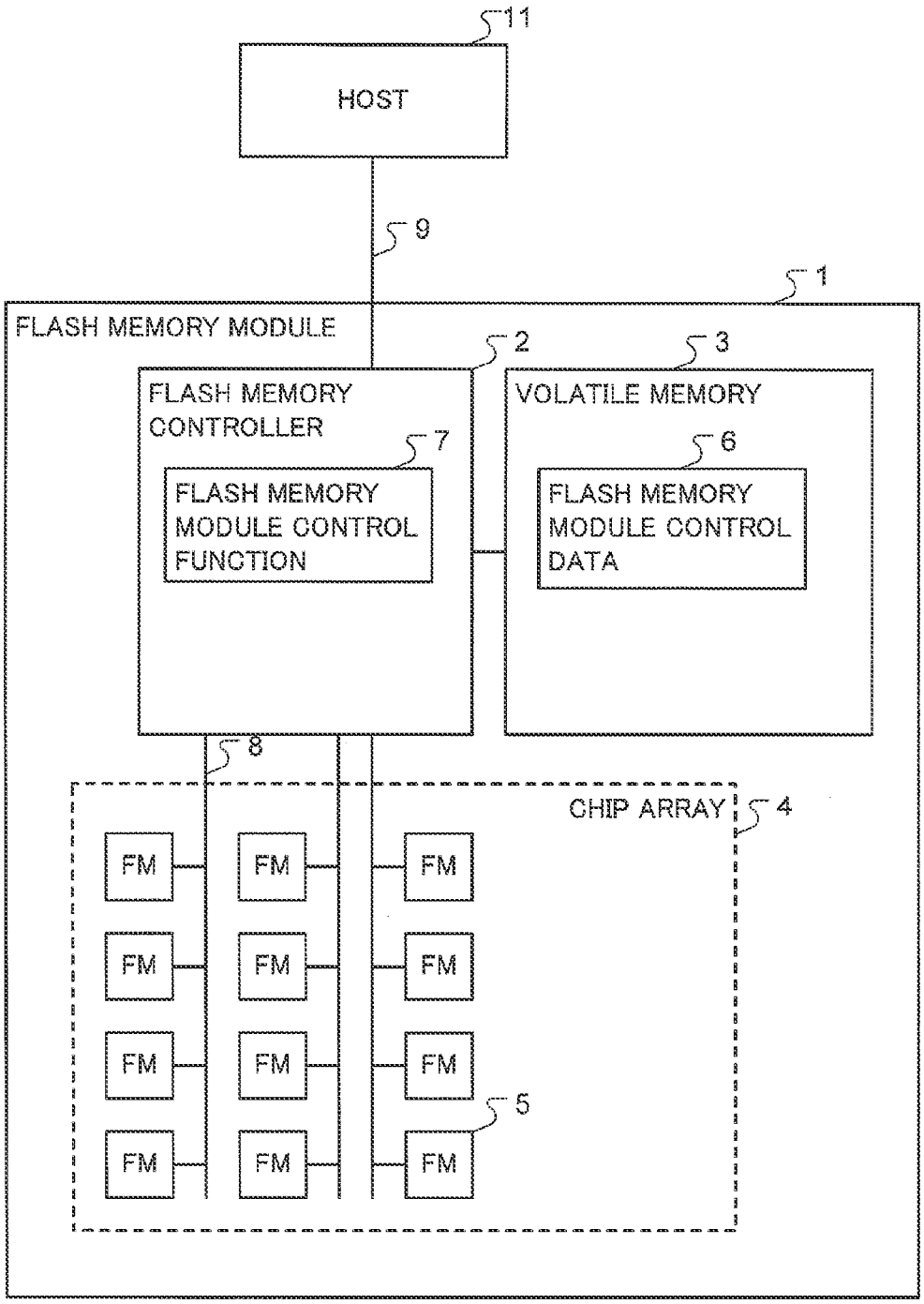

11
HOST
9
1
FLASH MEMORY MODULE
2
3
FLASH MEMORY CONTROLLER
7
FLASH MEMORY MODULE CONTROL FUNCTION
VOLATILE MEMORY
6
FLASH MEMORY MODULE CONTROL DATA
8
CHIP ARRAY
4
FM
5

FIG.1

[Fig. 2]

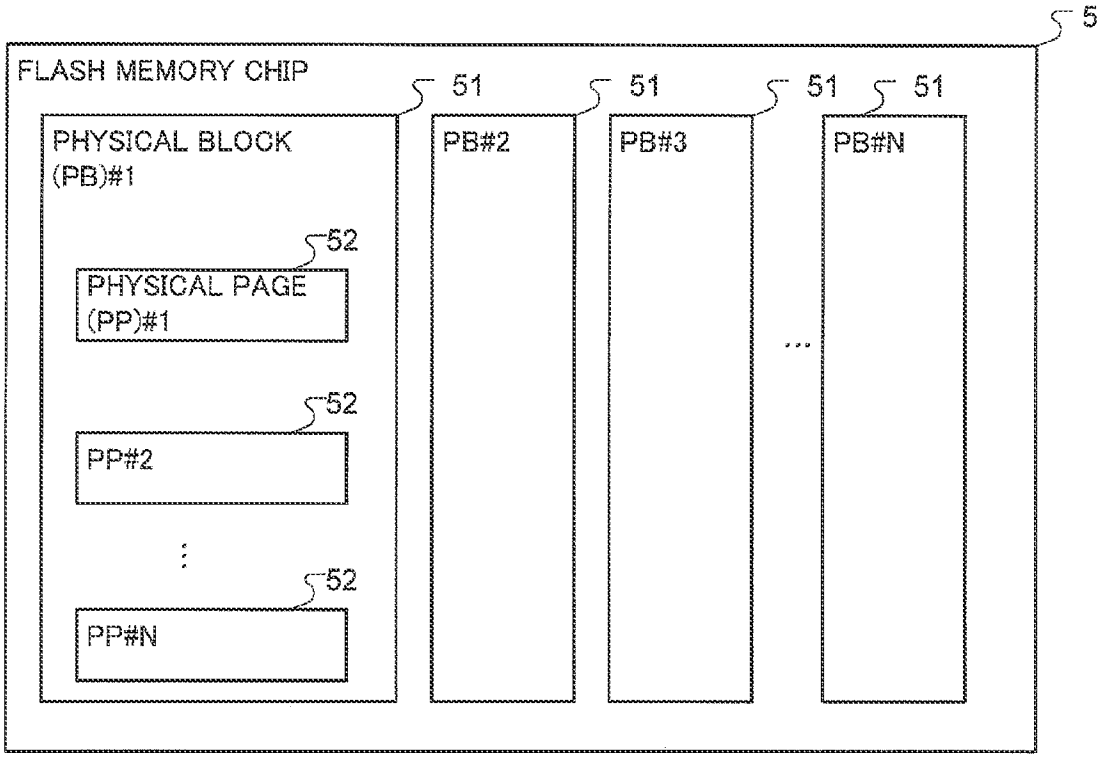


FIG.2

[Fig. 3A]

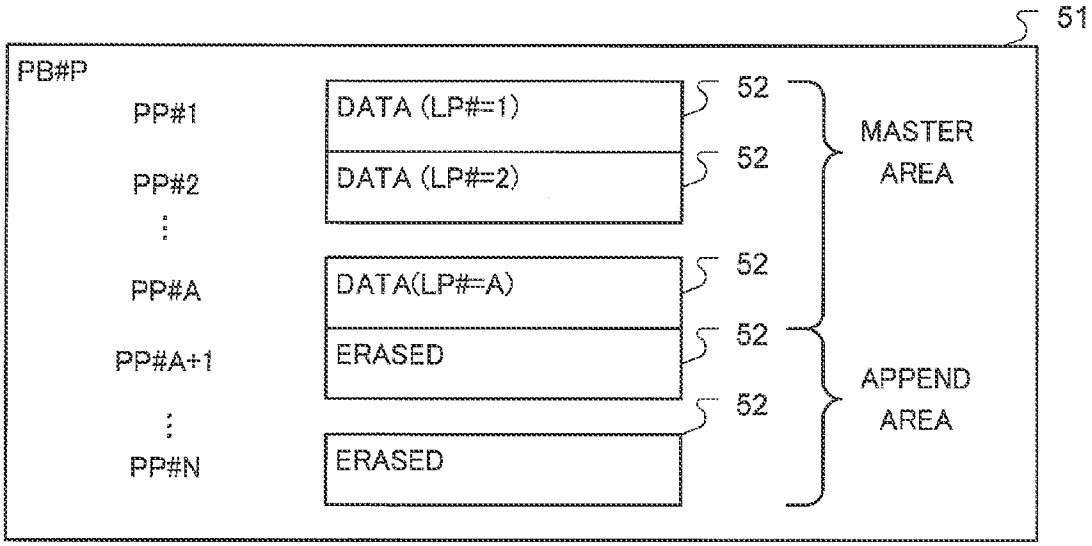


FIG.3A

[Fig. 3B]

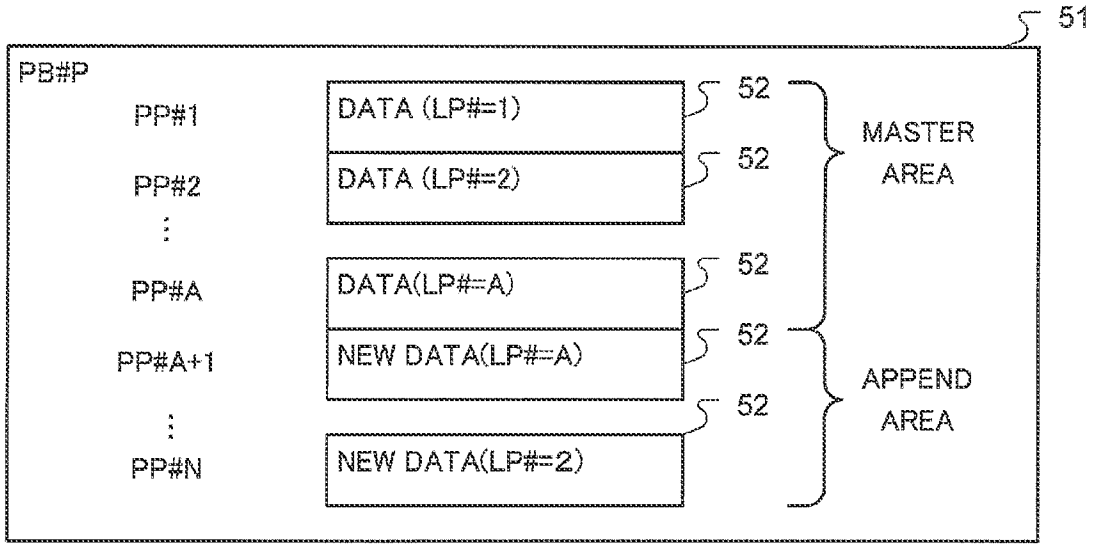


FIG.3B

[Fig. 3C]

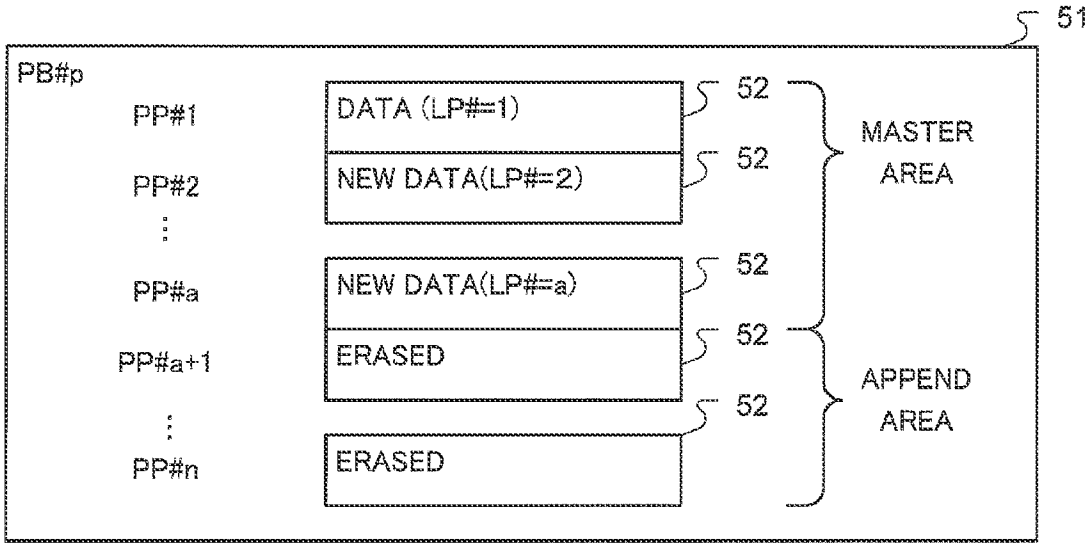


Fig.3C

[Fig. 4]

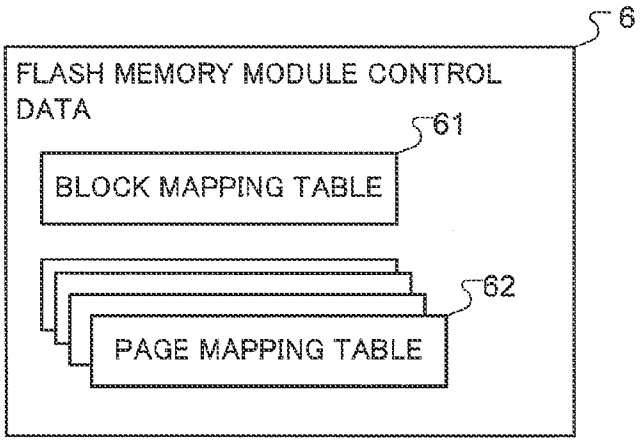


FIG.4

[Fig. 5]

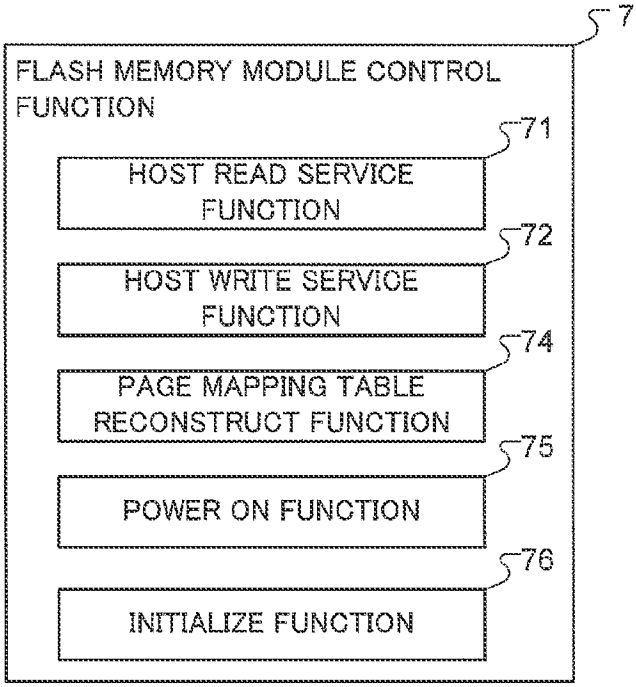


FIG.5

[Fig. 6]

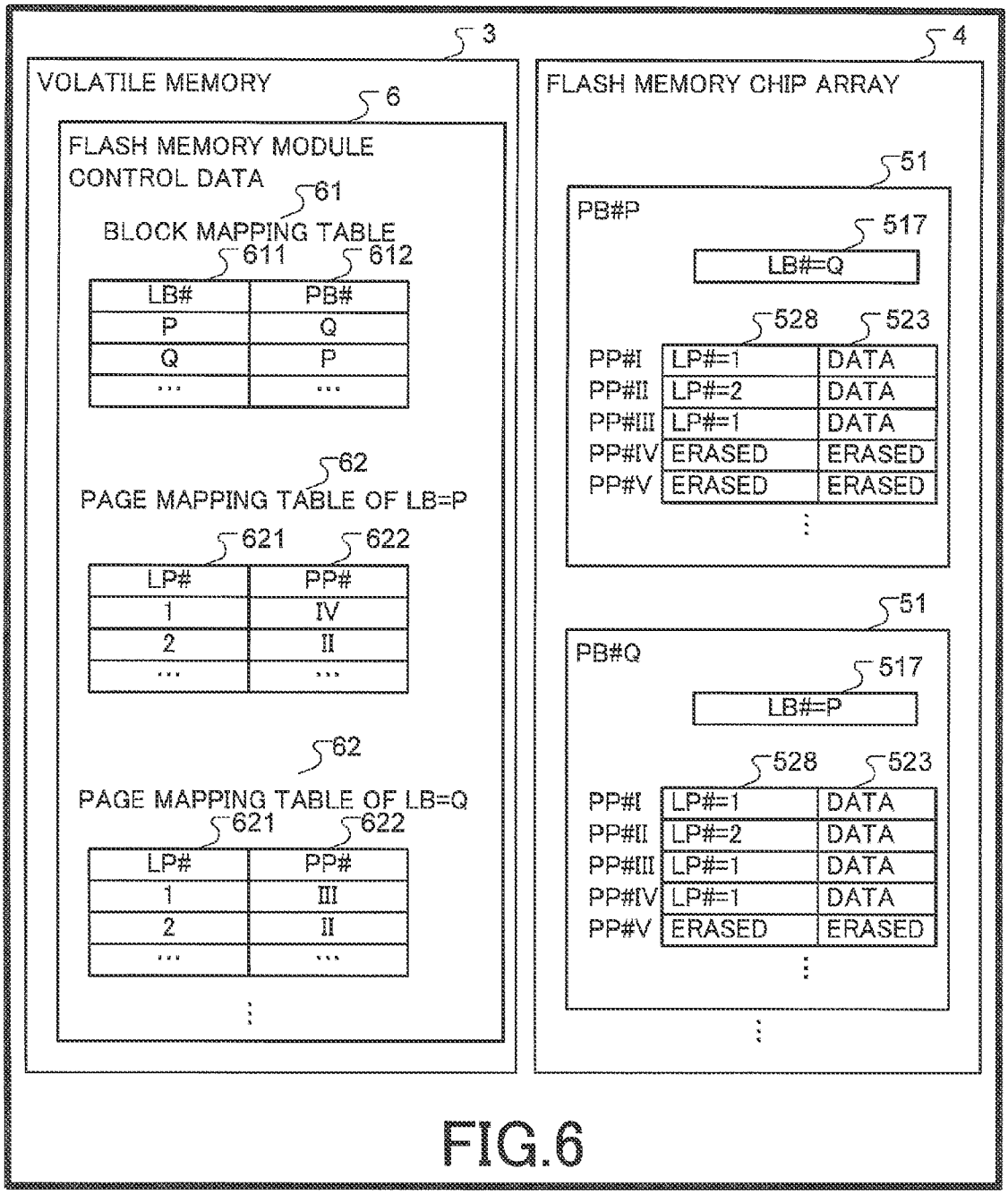

3
VOLATILE MEMORY
6
FLASH MEMORY MODULE CONTROL DATA
61
BLOCK MAPPING TABLE
611
612
LB#
PB#
P
Q
Q
P
62
PAGE MAPPING TABLE OF LB=P
621
622
LP#
PP#
1
IV
2
II
62
PAGE MAPPING TABLE OF LB=Q
621
622
LP#
PP#
1
III
2
II
4
FLASH MEMORY CHIP ARRAY
51
PB#P
517
LB#=Q
528
523
PP#I LP#=1 DATA
PP#II LP#=2 DATA
PP#III LP#=1 DATA
PP#IV ERASED ERASED
PP#V ERASED ERASED
51
PB#Q
517
LB#=P
528
523
PP#I LP#=1 DATA
PP#II LP#=2 DATA
PP#III LP#=1 DATA
PP#IV LP#=1 DATA
PP#V ERASED ERASED

FIG.6

[Fig. 7]

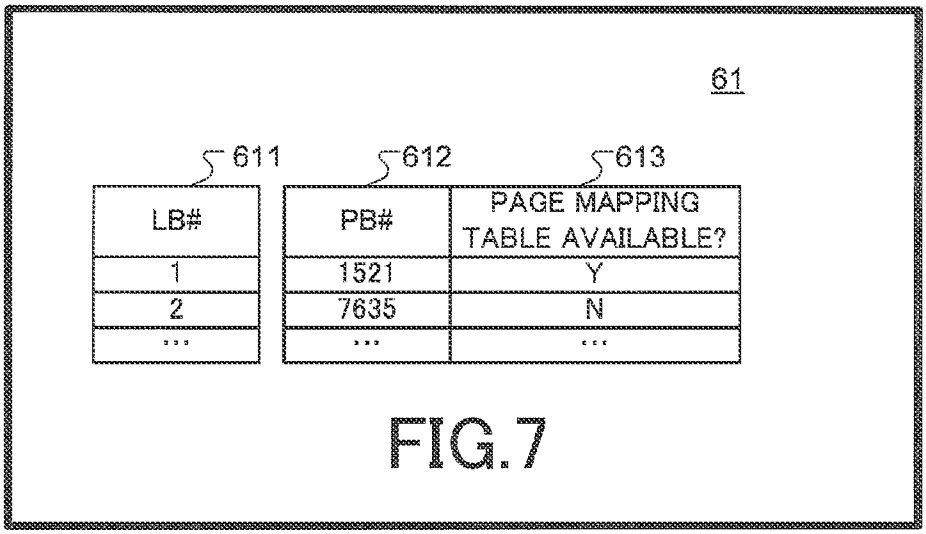


FIG.7

[Fig. 8]

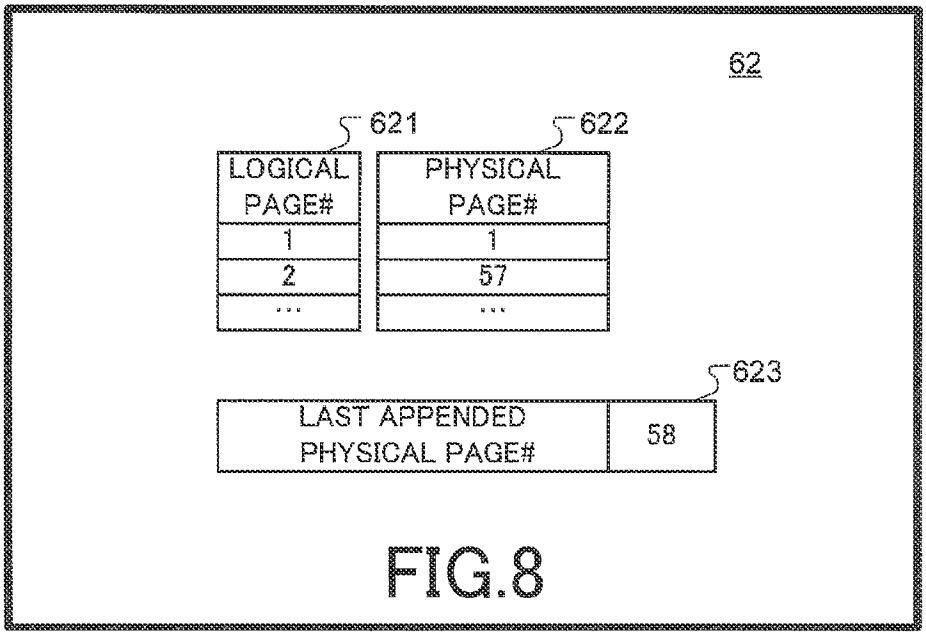


FIG.8

[Fig. 9]

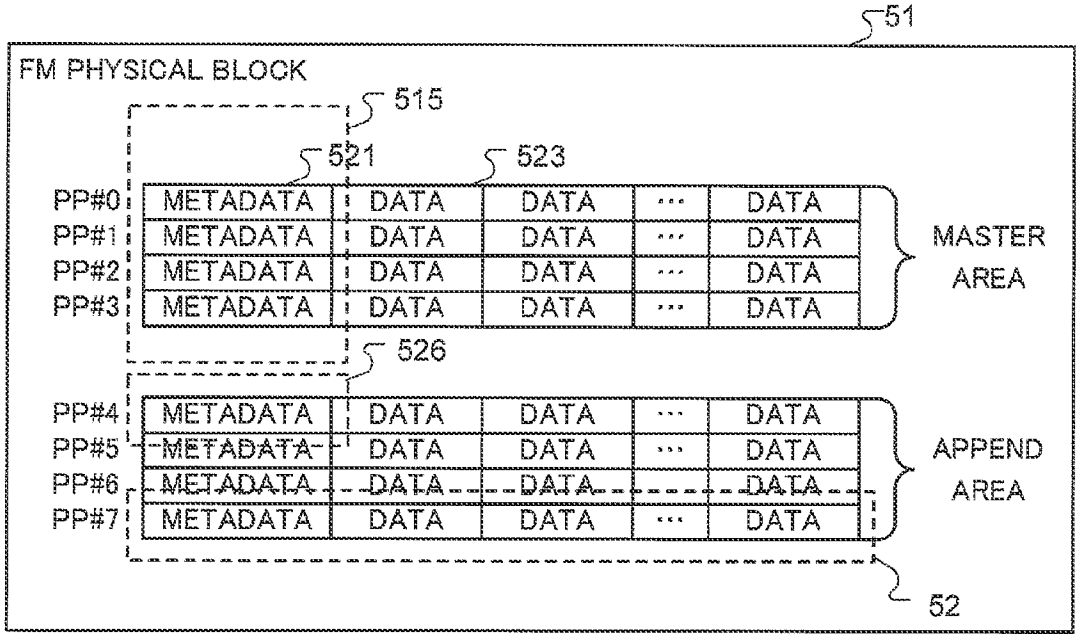


FIG.9

[Fig. 10]

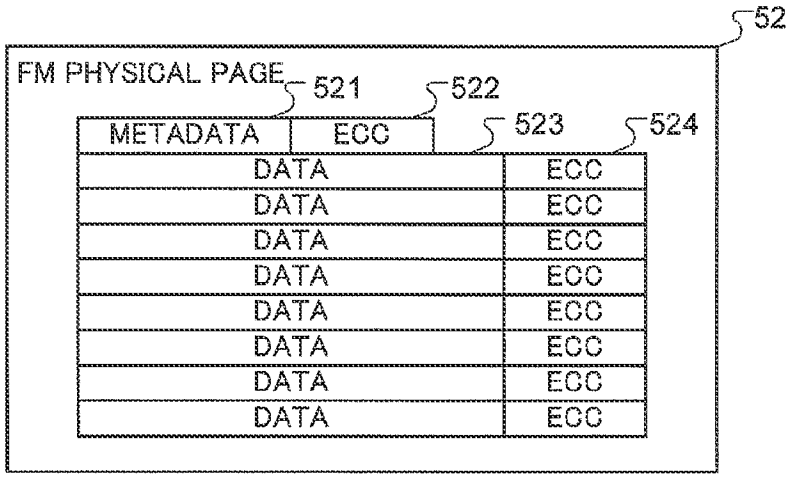


FIG.10

[Fig. 11]

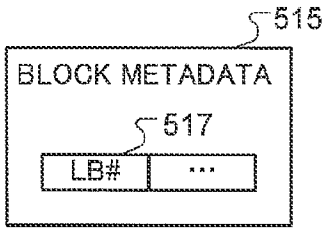


FIG.11

[Fig. 12]

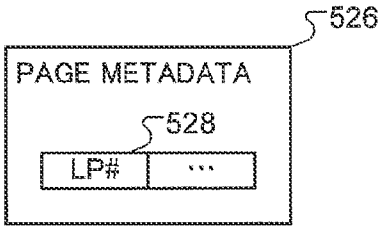


FIG.12

[Fig. 13]

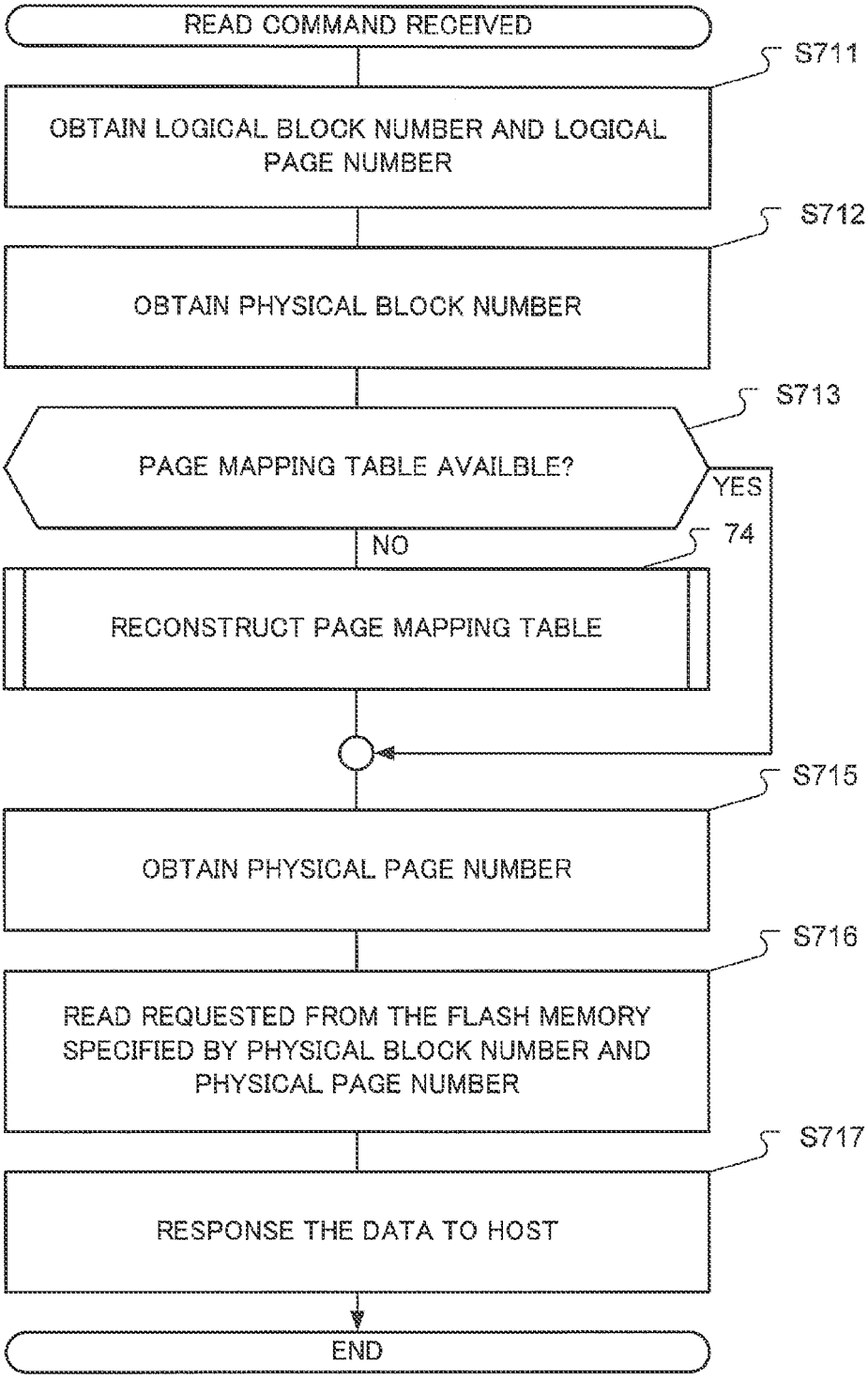

READ COMMAND RECEIVED
S711
OBTAIN LOGICAL BLOCK NUMBER AND LOGICAL PAGE NUMBER
S712
OBTAIN PHYSICAL BLOCK NUMBER
S713
PAGE MAPPING TABLE AVAILBLE?
YES
NO
74
RECONSTRUCT PAGE MAPPING TABLE
S715
OBTAIN PHYSICAL PAGE NUMBER
S716
READ REQUESTED FROM THE FLASH MEMORY SPECIFIED BY PHYSICAL BLOCK NUMBER AND PHYSICAL PAGE NUMBER
S717
RESPONSE THE DATA TO HOST
END
71

FIG.13

[Fig. 14]

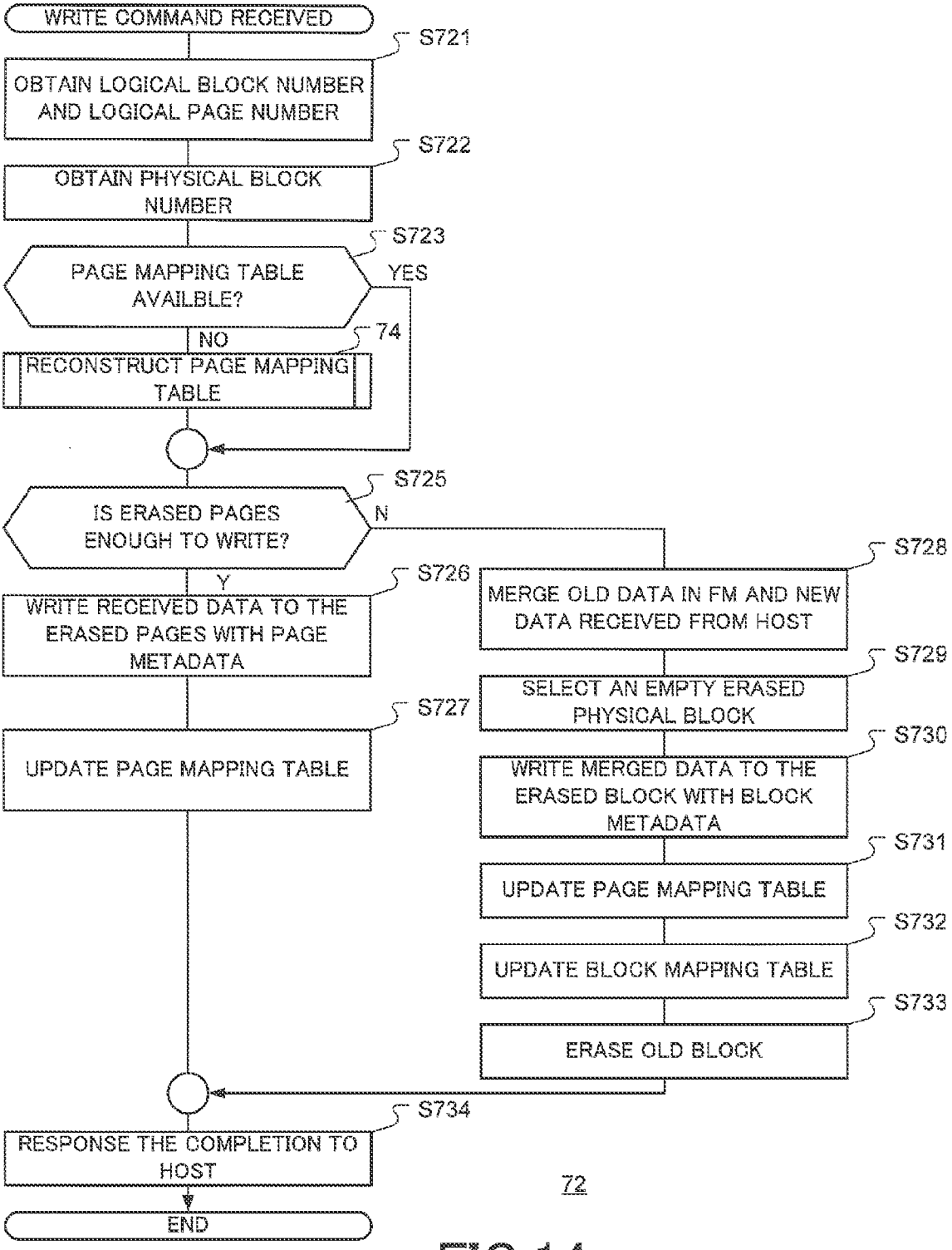

WRITE COMMAND RECEIVED
S721
OBTAIN LOGICAL BLOCK NUMBER AND LOGICAL PAGE NUMBER
S722
OBTAIN PHYSICAL BLOCK NUMBER
S723
PAGE MAPPING TABLE AVAILBLE?
YES
NO
74
RECONSTRUCT PAGE MAPPING TABLE
S725
IS ERASED PAGES ENOUGH TO WRITE?
N
Y
S726
WRITE RECEIVED DATA TO THE ERASED PAGES WITH PAGE METADATA
S727
UPDATE PAGE MAPPING TABLE
S728
MERGE OLD DATA IN FM AND NEW DATA RECEIVED FROM HOST
S729
SELECT AN EMPTY ERASED PHYSICAL BLOCK
S730
WRITE MERGED DATA TO THE ERASED BLOCK WITH BLOCK METADATA
S731
UPDATE PAGE MAPPING TABLE
S732
UPDATE BLOCK MAPPING TABLE
S733
ERASE OLD BLOCK
S734
RESPONSE THE COMPLETION TO HOST
END
72

FIG.14

[Fig. 15]

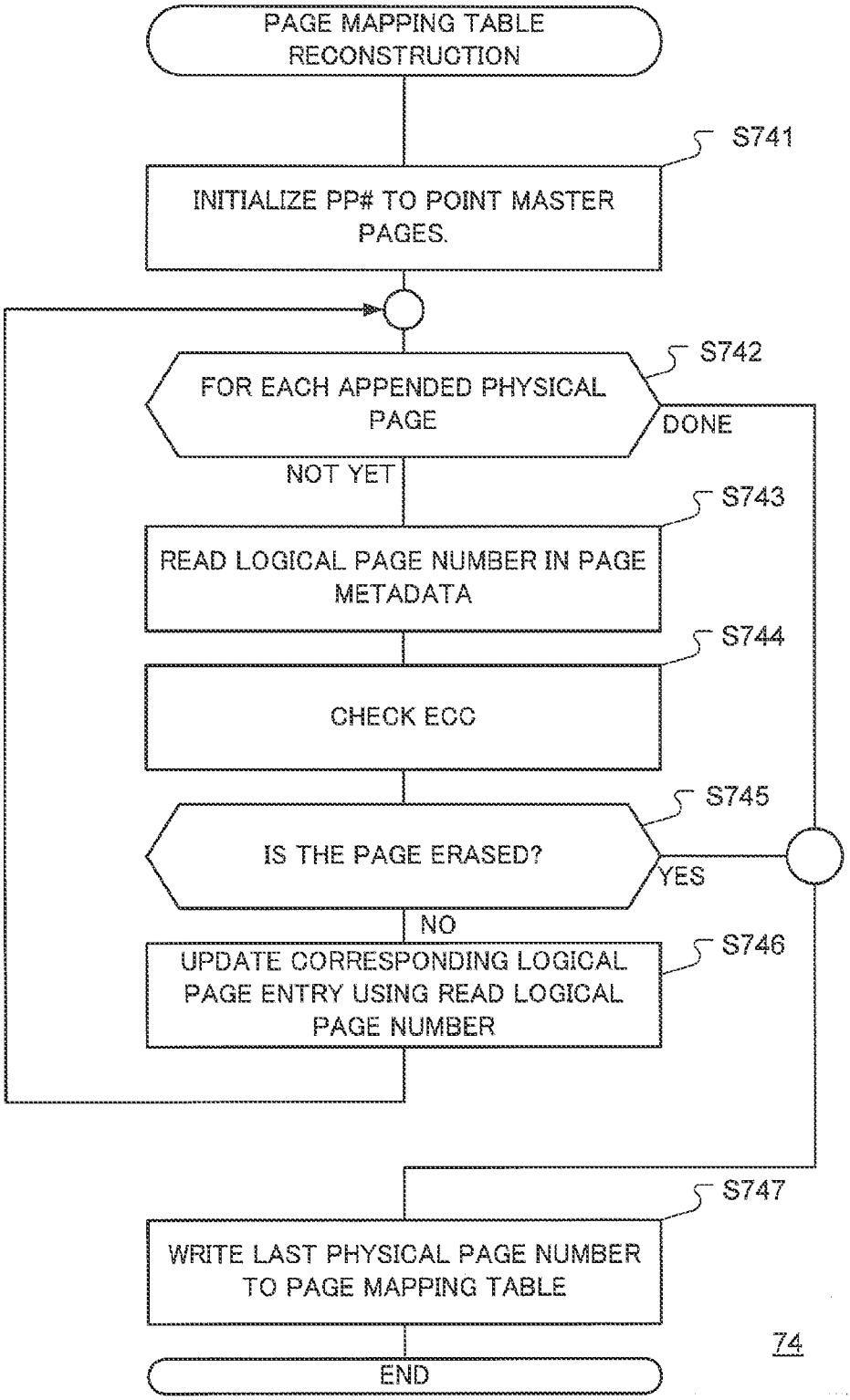

PAGE MAPPING TABLE RECONSTRUCTION
S741
INITIALIZE PP# TO POINT MASTER PAGES.
S742
FOR EACH APPENDED PHYSICAL PAGE
DONE
NOT YET
S743
READ LOGICAL PAGE NUMBER IN PAGE METADATA
S744
CHECK ECC
S745
IS THE PAGE ERASED?
YES
NO
S746
UPDATE CORRESPONDING LOGICAL PAGE ENTRY USING READ LOGICAL PAGE NUMBER
S747
WRITE LAST PHYSICAL PAGE NUMBER TO PAGE MAPPING TABLE
END
74

FIG.15

[Fig. 16]

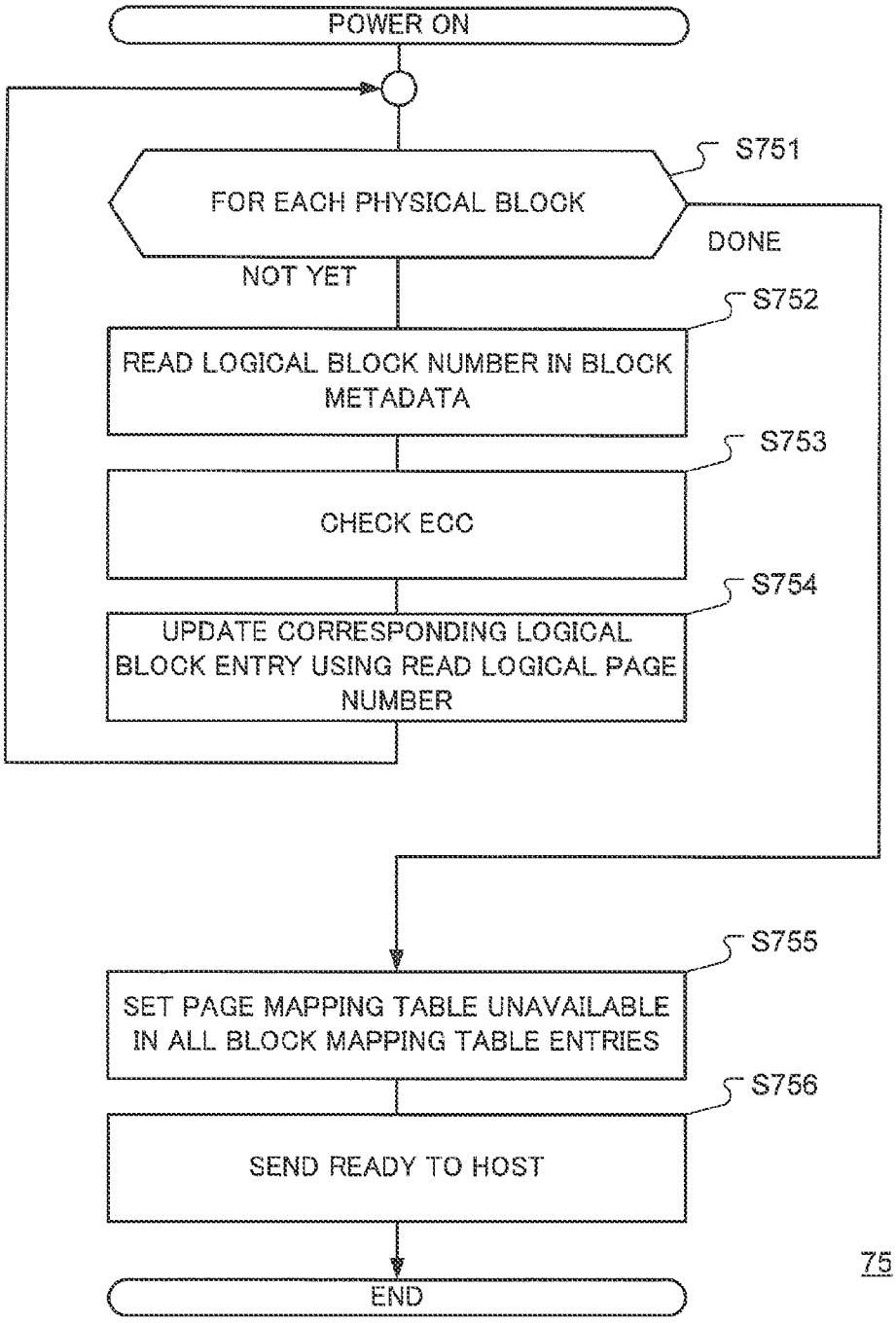

POWER ON
S751
FOR EACH PHYSICAL BLOCK
DONE
NOT YET
S752
READ LOGICAL BLOCK NUMBER IN BLOCK METADATA
S753
CHECK ECC
S754
UPDATE CORRESPONDING LOGICAL BLOCK ENTRY USING READ LOGICAL PAGE NUMBER
S755
SET PAGE MAPPING TABLE UNAVAILABLE IN ALL BLOCK MAPPING TABLE ENTRIES
S756
SEND READY TO HOST
END
75

FIG.16

[Fig. 17]

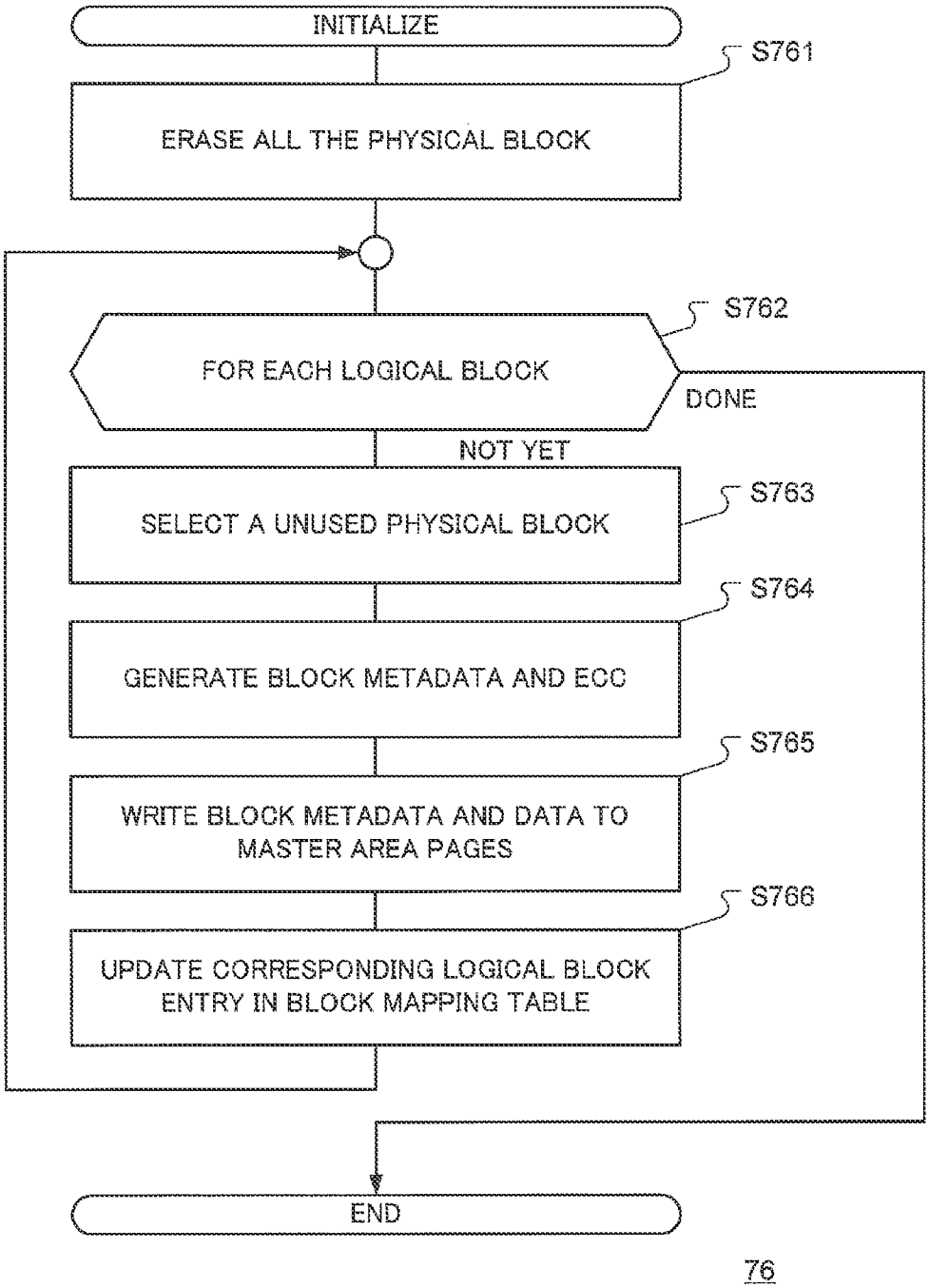

INITIALIZE
S761
ERASE ALL THE PHYSICAL BLOCK
S762
FOR EACH LOGICAL BLOCK
DONE
NOT YET
S763
SELECT A UNUSED PHYSICAL BLOCK
S764
GENERATE BLOCK METADATA AND ECC
S765
WRITE BLOCK METADATA AND DATA TO MASTER AREA PAGES
S766
UPDATE CORRESPONDING LOGICAL BLOCK ENTRY IN BLOCK MAPPING TABLE
END
76

FIG.17

[Fig. 18]

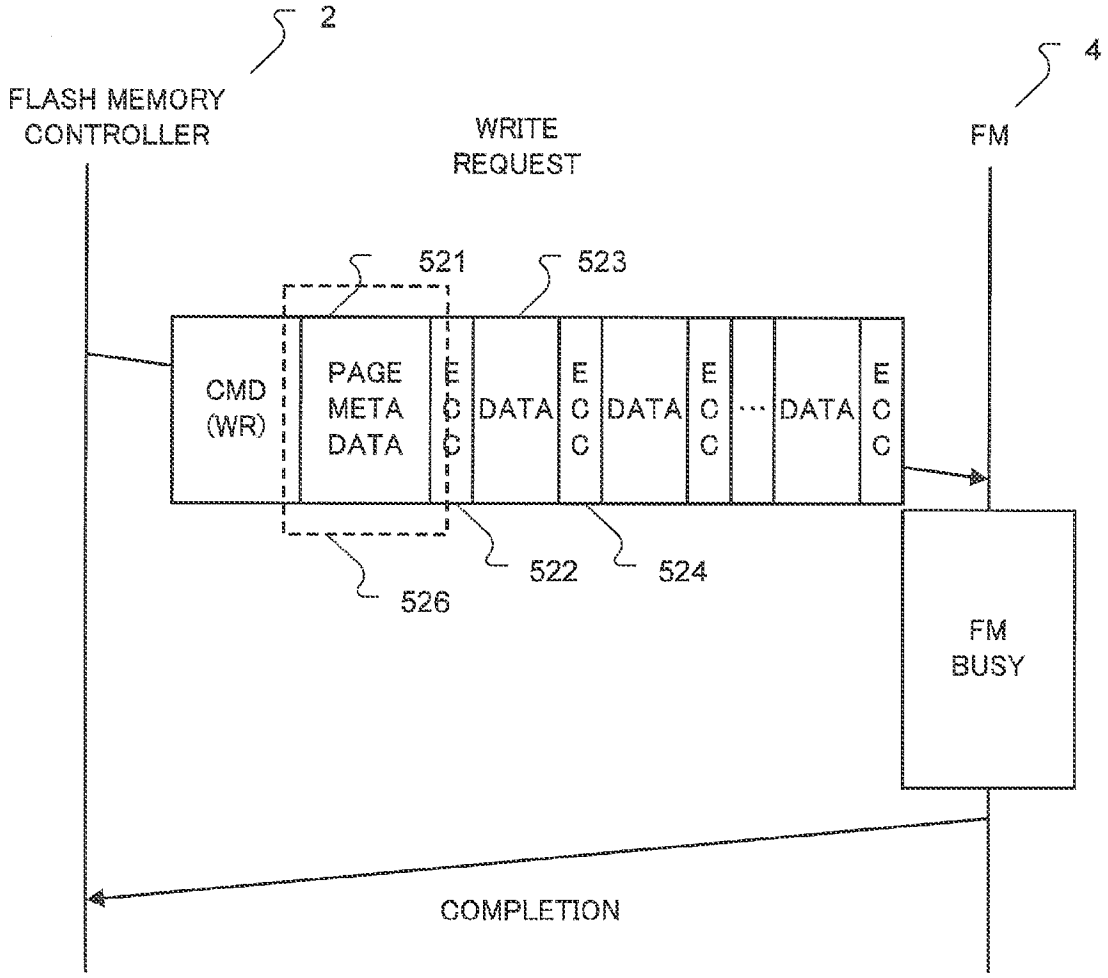

2
FLASH MEMORY CONTROLLER
WRITE REQUEST
4
FM
521
523
CMD (WR)
PAGE META DATA
ECC
DATA
ECC
DATA
ECC
...
DATA
ECC
522
524
526
FM BUSY
COMPLETION

FIG.18

[Fig. 19]

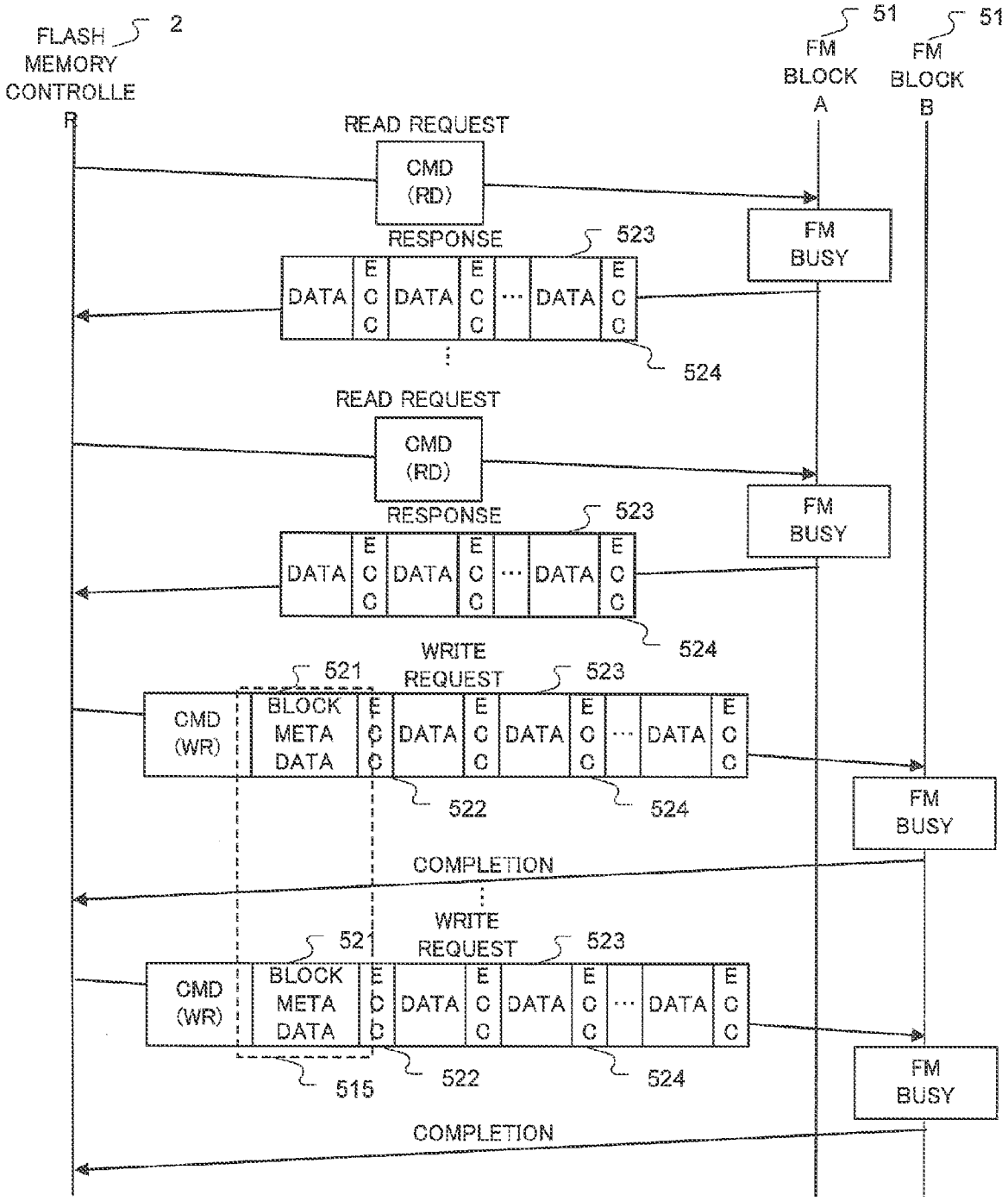

FLASH MEMORY CONTROLLER
2
FM BLOCK A
51
FM BLOCK B
51
READ REQUEST
CMD (RD)
FM BUSY
RESPONSE
523
DATA
ECC
524
READ REQUEST
CMD (RD)
FM BUSY
RESPONSE
523
WRITE REQUEST
521
523
CMD (WR)
BLOCK META DATA
522
524
FM BUSY
COMPLETION
WRITE REQUEST
521
523
CMD (WR)
515
522
524
FM BUSY
COMPLETION

FIG.19

[Fig. 20]

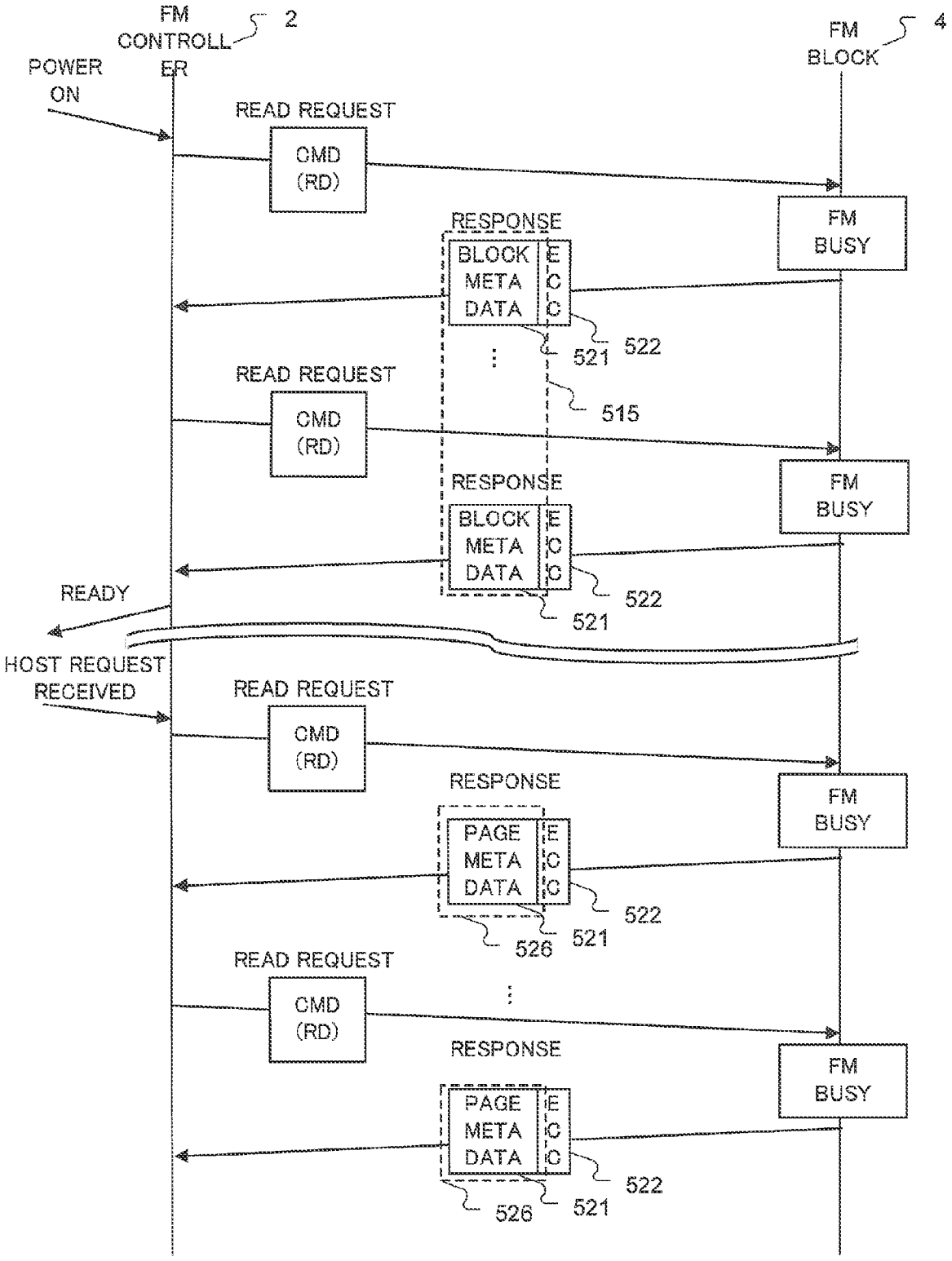

FM CONTROLLER
2
FM BLOCK
4
POWER ON
READ REQUEST
CMD (RD)
FM BUSY
RESPONSE
BLOCK META DATA
ECC
521
522
515
READY
HOST REQUEST RECEIVED
PAGE META DATA
526

FIG.20

[Fig. 21]

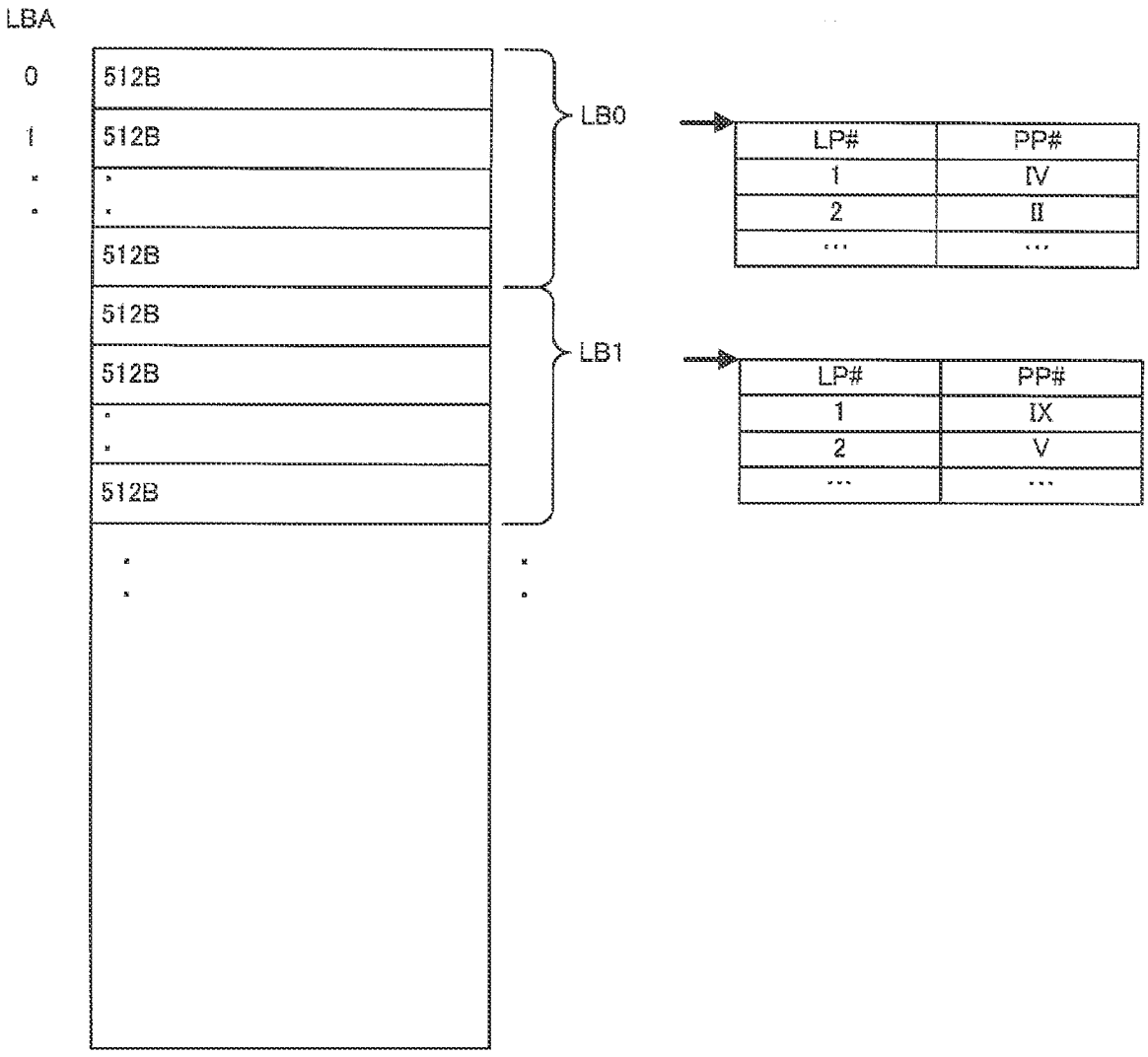

LBA
0
1
512B
512B
512B
512B
512B
512B
512B
LB0
LB1
LP#
PP#
1
IV
2
II
LP#
PP#
1
IX
2
V

FIG.21

INTERNATIONAL SEARCH REPORT

International application No
PCT/JP2009/005859

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F12/02
ADD.

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

B. FIELDS SEARCHED

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

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

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

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2004/109376 A1 (LIN JIN-SHIN [TW]) 10 June 2004 (2004-06-10) paragraph [0009] - paragraph [0011]	1,6-15
X	US 2007/300037 A1 (ROGERS ANDREW [US] ET AL) 27 December 2007 (2007-12-27) paragraph [0005] - paragraph [0007] figure 1	1,12,15
A	EP 1 939 750 A1 (NETAC TECHNOLOGY CO LTD [CN]) 2 July 2008 (2008-07-02) paragraph [0007] - paragraph [0010]	1-15

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

☒ See patent family annex.

* Special categories of cited documents :

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

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

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
13 July 2010	21/07/2010

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Knutsson, Frédéric

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/JP2009/005859

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2004109376 A1	10-06-2004	NONE	
US 2007300037 A1	27-12-2007	NONE	
EP 1939750 A1	02-07-2008	WO 2007033614 A1	29-03-2007
		CN 1936867 A	28-03-2007
		JP 2009510552 T	12-03-2009
		KR 20080067334 A	18-07-2008
		US 2009313417 A1	17-12-2009
		ZA 200803637 A	28-10-2009

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

20995 7590 01/10/2023

KNOBBE MARTENS OLSON & BEAR LLP
2040 MAIN STREET
FOURTEENTH FLOOR
IRVINE, CA 92614

EXAMINER
BATAILLE, PIERRE MICHE

ART UNIT	PAPER NUMBER
2136	

DATE MAILED: 01/10/2023

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/931,506	09/12/2022	Andrey V. Kuzmin	RMMS.003C12	3894

TITLE OF INVENTION: STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$480	$0.00	$0.00	$480	04/10/2023

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 ENTITY STATUS shown above. If the ENTITY STATUS is shown as SMALL or MICRO, verify whether entitlement to that entity status still applies.

If the ENTITY STATUS is the same as shown above, pay the TOTAL FEE(S) DUE shown above.

If the ENTITY STATUS is changed from that shown above, on PART B - FEE(S) TRANSMITTAL, complete section number 5 titled "Change in Entity Status (from status indicated above)".

For purposes of this notice, small entity fees are 40% the amount of undiscounted fees, and micro entity fees are 20% the amount of undiscounted fees.

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. 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: Maintenance fees are due in utility patents issuing on applications filed on or after Dec. 12, 1980. It is patentee's responsibility to ensure timely payment of maintenance fees when due. More information is available at www.uspto.gov/PatentMaintenanceFees.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), by mail or fax, or via EFS-Web.

By mail, send to: Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

By fax, send to: (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)

20995 7590 01/10/2023

KNOBBE MARTENS OLSON & BEAR LLP
2040 MAIN STREET
FOURTEENTH FLOOR
IRVINE, CA 92614

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 transmitted to the USPTO via EFS-Web or by facsimile to (571) 273-2885, on the date below.

(Typed or printed name)
(Signature)
(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/931,506	09/12/2022	Andrey V. Kuzmin	RMMS.003C12	3894

TITLE OF INVENTION: STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$480	$0.00	$0.00	$480	04/10/2023

EXAMINER	ART UNIT	CLASS-SUBCLASS
BATAILLE, PIERRE MICHE	2136	710-301000

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/AIA/122 or PTO/SB/122) attached.

❑ "Fee Address" indication (or "Fee Address" Indication form PTO/AIA/47 or 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 must have been previously recorded, or filed for recordation, as set forth in 37 CFR 3.11 and 37 CFR 3.81(a). 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. Fees submitted: ❑ Issue Fee ❑ Publication Fee (if required) ❑ Advance Order - # of Copies _____

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

❑ Electronic Payment via EFS-Web ❑ Enclosed check ❑ Non-electronic payment by credit card (Attach form PTO-2038)

❑ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment to Deposit Account No. _____

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

❑ Applicant certifying micro entity status. See 37 CFR 1.29

❑ Applicant asserting small entity status. See 37 CFR 1.27

❑ Applicant changing to regular undiscounted fee status.

NOTE: Absent a valid certification of Micro Entity Status (see forms PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.
NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.
NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: This form must be signed in accordance with 37 CFR 1.31 and 1.33. See 37 CFR 1.4 for signature requirements and certifications.

Authorized Signature _____ Date _____

Typed or printed name _____ Registration No. _____

SAMSUNG EX1002, Page 193 of 1031

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.
17/931,506	09/12/2022	Andrey V. Kuzmin	RMMS.003C12	3894

20995 7590 01/10/2023

KNOBBE MARTENS OLSON & BEAR LLP
2040 MAIN STREET
FOURTEENTH FLOOR
IRVINE, CA 92614

EXAMINER	
BATAILLE, PIERRE MICHE	
ART UNIT	PAPER NUMBER
2136	

DATE MAILED: 01/10/2023

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

The Office has discontinued providing a Patent Term Adjustment (PTA) calculation with the Notice of Allowance.

Section 1(h)(2) of the AIA Technical Corrections Act amended 35 U.S.C. 154(b)(3)(B)(i) to eliminate the requirement that the Office provide a patent term adjustment determination with the notice of allowance. See Revisions to Patent Term Adjustment, 78 Fed. Reg. 19416, 19417 (Apr. 1, 2013). Therefore, the Office is no longer providing an initial patent term adjustment determination with the notice of allowance. The Office will continue to provide a patent term adjustment determination with the Issue Notification Letter that is mailed to applicant approximately three weeks prior to the issue date of the patent, and will include the patent term adjustment on the patent. Any request for reconsideration of the patent term adjustment determination (or reinstatement of patent term adjustment) should follow the process outlined in 37 CFR 1.705.

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.

OMB Clearance and PRA Burden Statement for PTOL-85 Part B

The Paperwork Reduction Act (PRA) of 1995 requires Federal agencies to obtain Office of Management and Budget approval before requesting most types of information from the public. When OMB approves an agency request to collect information from the public, OMB (i) provides a valid OMB Control Number and expiration date for the agency to display on the instrument that will be used to collect the information and (ii) requires the agency to inform the public about the OMB Control Number's legal significance in accordance with 5 CFR 1320.5(b).

The information collected by PTOL-85 Part B 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 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, 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.

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Notice of Allowability	Application No. 17/931,506	Applicant(s) Kuzmin et al.	
	Examiner PIERRE MICHEL BATAILLE	Art Unit 2136	AIA (FITF) Status Yes

-- 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 12/13/2022.
   ☐ A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on _____.

2. ☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on _____; the restriction requirement and election have been incorporated into this action.

3. ☑ The allowed claim(s) is/are 2-31 . As a result of the allowed claim(s), you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see **http://www.uspto.gov/patents/init_events/pph/index.jsp** or send an inquiry to **PPHfeedback@uspto.gov.**

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

   Certified copies:

   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.

5. ☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
   ☐ 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. ☑ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date 12/13/2022.
3. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material _____.
4. ☐ Interview Summary (PTO-413), Paper No./Mail Date. _____.
5. ☐ Examiner's Amendment/Comment
6. ☑ Examiner's Statement of Reasons for Allowance
7. ☐ Other _____.

/PIERRE MICHEL BATAILLE/
Primary Examiner, Art Unit 2136

Notice of Pre-AIA or AIA Status

The present application, filed on or after March 16, 2013, is being examined under the first inventor to file provisions of the AIA.

Allowable Subject Matter

Claims 2-31 are allowed.

The following is an examiner's statement of reasons for allowance:

The present invention features:

storage device comprising: flash memory divided into zones, each zone of the zones being a subdivision comprising a set of erase units, and each erase unit of the set of erase units comprising host accessible pages, wherein each zone: is associated with a logical address range, has a zone size for which the storage device is operable to receive a management command from a host that results in the storage device performing an erasure of one or more erase units of the set of erase units

Related prior art include:

WO2008057557A2 (WARE et al) teaching memory system supporting non-volatile memory that exhibits wear mechanism asymmetrically associated with write operations.

WO 2005055043 A1 (KARR et al) teaching off-host virtualizer of a system and method for emulating operating system metadata, the virtualizer configured to

generate operating system metadata for a first virtual storage device and to make the operating system metadata accessible to the first host, wherein the metadata is customized in accordance with a requirement of the operating system in use at the first host.

EP 1157328 B1 (SINCLAIR et al) teaching memory system having a solid state memory comprising non-volatile individually addressable memory sectors arranged in erasable blocks, and a controller for writing to reading from the sectors, and for sorting the blocks into 'erased' and 'not erased' blocks, the controller performing logical to physical address translation, and includes a Write Pointer (WP) for pointing to the physical sector address to which data is to be written from a host processor.

US 2006/0106972 (GOROBETS et al) teaching re-programmable non-volatile memory system having its memory cells grouped into blocks of cells that are simultaneously erasable is operated in a manner to level out the wear of the individual blocks through repetitive erasing and re-programming.

US 2006/0184719 (SINCLAIR) teaching method of operating a re-programmable non-volatile mass storage system having memory cells organized into blocks that are erased prior to any new data being written therein and which individually have a capacity of storing multiple units of data.

However, none of the prior art in teaches or renders obvious the features as recited in the claims, taken in light of the disclosure, specifically as detailed on Par. 0183-0184.

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

Contact Information

Any inquiry concerning this communication or earlier communications from the examiner should be directed to PIERRE MICHEL BATAILLE whose telephone number is (571)272-4178. The examiner can normally be reached Monday - Thursday 7-6 ET.

Examiner interviews are available via telephone, in-person, and video conferencing using a USPTO supplied web-based collaboration tool. To schedule an interview, applicant is encouraged to use the USPTO Automated Interview Request (AIR) at http://www.uspto.gov/interviewpractice.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Charles Rones can be reached on (571) 272-4085. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of published or unpublished applications may be obtained from Patent Center. Unpublished application information in Patent Center is available to registered users. To file and manage patent submissions in Patent Center, visit: https://patentcenter.uspto.gov. Visit https://www.uspto.gov/patents/apply/patent-center for more information about Patent Center and https://www.uspto.gov/patents/docx for information about filing in DOCX format. For additional questions, contact the Electronic

Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO

Customer Service Representative, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/PIERRE MICHEL BATAILLE/
Primary Examiner, Art Unit 2136

Notice of References Cited	Application/Control No. 17/931,506	Applicant(s)/Patent Under Reexamination Kuzmin et al.	
	Examiner PIERRE MICHEL BATAILLE	Art Unit 2136	Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	CPC Classification	US Classification
*	A	US-20060184719-A1	08-2006	Sinclair; Alan W.	G06F3/0679	711/E12.008
*	B	US-20060106972-A1	05-2006	Gorobets; Sergey A.	G06F12/0246	711/E12.008
	C					
	D					
	E					
	F					
	G					
	H					
	I					
	J					
	K					
	L					
	M					

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	CPC Classification
	N	WO-2008057557-A2	05-2008	WO	TSERN E K	G06F12/0246
	O	WO-2005055043-A1	06-2005	WO	KARR R	G06F3/0607
	P	EP-1157328-B1	05-2005	EP	GOROBETS A	G06F12/0246
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/931,506	Kuzmin et al.
	Examiner	**Art Unit**
	PIERRE MICHEL BATAILLE	2136

CPC						
Symbol					**Type**	**Version**
G06F	/	12	/	0246	F	2013-01-01
G06F	/	16	/	1847	I	2019-01-01
G06F	/	11	/	1068	I	2013-01-01
G06F	/	3	/	0611	I	2013-01-01
G06F	/	12	/	10	I	2013-01-01
G06F	/	8	/	654	I	2018-02-01
G06F	/	3	/	0679	I	2013-01-01
G06F	/	3	/	0638	I	2013-01-01
G06F	/	3	/	0688	I	2013-01-01
G06F	/	11	/	1072	I	2013-01-01
G06F	/	2015	/	766	A	2013-01-01
G06F	/	2212	/	7208	A	2013-01-01
G06F	/	2212	/	2022	A	2013-01-01
G06F	/	2212	/	7207	A	2013-01-01

CPC Combination Sets							
Symbol					**Type**	**Set**	**Ranking**	**Version**
	/		/					

NONE		**Total Claims Allowed:**	
(Assistant Examiner)	(Date)	30	
/PIERRE MICHEL BATAILLE/ Primary Examiner, Art Unit 2136	21 December 2022	O.G. Print Claim(s)	O.G. Print Figure
(Primary Examiner)	(Date)	1	15

U.S. Patent and Trademark Office — Part of Paper No.: 20221221

SAMSUNG EX1002, Page 202 of 1031

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/931,506	Kuzmin et al.
	Examiner	**Art Unit**
	PIERRE MICHEL BATAILLE	2136

INTERNATIONAL CLASSIFICATION					
CLAIMED					
G06F	/	12	/	02	
G06F	/	12	/	10	

NON-CLAIMED				
	/		/	

US ORIGINAL CLASSIFICATION	
CLASS	**SUBCLASS**

CROSS REFERENCES(S)						
CLASS	**SUBCLASS (ONE SUBCLASS PER BLOCK)**					

NONE		Total Claims Allowed:	
(Assistant Examiner)	(Date)	30	
/PIERRE MICHEL BATAILLE/ Primary Examiner, Art Unit 2136	21 December 2022	O.G. Print Claim(s)	O.G. Print Figure
(Primary Examiner)	(Date)	1	15

U.S. Patent and Trademark Office — Part of Paper No.: 20221221

SAMSUNG EX1002, Page 203 of 1031

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/931,506	Kuzmin et al.
	Examiner	**Art Unit**
	PIERRE MICHEL BATAILLE	2136

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

CLAIMS

Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original
	1	9	10	18	19	27	28								
1	2	10	11	19	20	28	29								
2	3	11	12	20	21	29	30								
3	4	12	13	21	22	30	31								
4	5	13	14	22	23										
5	6	14	15	23	24										
6	7	15	16	24	25										
7	8	16	17	25	26										
8	9	17	18	26	27										

NONE		**Total Claims Allowed:**	
(Assistant Examiner)	(Date)	30	
/PIERRE MICHEL BATAILLE/ Primary Examiner, Art Unit 2136	21 December 2022	O.G. Print Claim(s)	O.G. Print Figure
(Primary Examiner)	(Date)	1	15

U.S. Patent and Trademark Office

Part of Paper No.: 20221221

SAMSUNG EX1002, Page 204 of 1031

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/931,506	Kuzmin et al.
	Examiner	Art Unit
	PIERRE MICHEL BATAILLE	2136

CPC - Searched*		
Symbol	Date	Examiner
G06F12/0246, G06F12/10, G06F12/02, G06F3/0611, G06F3/0679, G06F3/0638	12/21/2022	PMB

CPC Combination Sets - Searched*		
Symbol	Date	Examiner

US Classification - Searched*			
Class	Subclass	Date	Examiner

* See search history printout included with this form or the SEARCH NOTES box below to determine the scope of the search.

Search Notes		
Search Notes	Date	Examiner
PALM INVENTOR & ASSIGNEE NAME SEARCH	11/16/2022	PMB
KEYORD & CONTEXT PE2E SEARCH	11/16/2022	PMB
NON-PATENT LITERATURE SEARCH	11/16/2022	PMB
KEYORD & CONTEXT PE2E SEARCH	11/17/2022	PMB
NON-PATENT LITERATURE SEARCH	12/21/2022	PMB
KEYORD & CONTEXT PE2E SEARCH	12/21/2022	PMB

/PIERRE MICHEL BATAILLE/ Primary Examiner, Art Unit 2136	

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/931,506	Kuzmin et al.
	Examiner	Art Unit
	PIERRE MICHEL BATAILLE	2136

Interference Search			
US Class/CPC Symbol	**US Subclass/CPC Group**	**Date**	**Examiner**
	INTERFERENCE SEARCH INCLUDED IN WEST SEARCH	11/16/2022	PMB
	INTERFERENCE SEARCH INCLUDED IN WEST SEARCH		PMB

/PIERRE MICHEL BATAILLE/ Primary Examiner, Art Unit 2136	

PE2E SEARCH - Search History (Prior Art)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	British Equivalents	Time Stamp
L2	1	(lookup OR look?up) near3 table WITH eras$ near5 address$2 AND page near2 address$ WITH (preserv$ OR reserv$ OR sav$3)	(USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/17 10:45 AM
L3	5	(lookup OR look?up) near3 table WITH eras$ near5 address$2 AND page near2 address$ WITH (preserv$ OR reserv$ OR sav$3)	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/17 11:04 AM
L4	16	(lookup OR look?up) near3 table WITH eras$ near5 address$2 AND address$ WITH (preserv$ OR reserv$ OR sav$3)	(USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/17 11:05 AM
L9	89	flash WITH controller AND interface WITH controller AND (map$ OR translat$) WITH physical WITH logical AND (translat$ OR map$) WITH page near3 address WITH (eras$ OR invalidat$) WITH table	(USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/17 03:15 PM
L10	60	flash WITH controller	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/17

		AND interface WITH controller AND (map$ OR translat$) WITH physical WITH logical AND (translat$ OR map$) WITH page near3 address WITH (eras$ OR invalidat$) WITH table					03:15 PM
L13	423	(maintain$ OR preserv$) WITH (block OR page) near3 address$ WITH (eras$ OR invalidat$)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/17 04:15 PM
L14	872	(maintain$ OR preserv$) WITH (block OR page) near3 address$ WITH (eras$ OR invalidat$)	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/17 04:15 PM
L25	153448	(G06F12/0246 G06F12/10 G06F12/02 G06F3/0611 G06F3/0679 G06F3/0638).CPC.	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/21 06:20 PM
L26	130	L25 AND ((unit OR zone OR block) AND maintain$ OR preserv$) WITH (block OR page OR unit OR zone) near3 address$ WITH eras$ near5 (page OR block)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN,	OR	ON	ON	2022/12/21 06:22 PM

			WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L27	62	L26 AND flash WITH controller AND interface WITH controller AND (map$ OR translat$) WITH physical WITH logical AND (translat$ OR map$) WITH (block OR page)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/21 06:22 PM
L28	335	L25 AND ((unit OR zone OR block) AND maintain$ OR preserv$) WITH (block OR page OR unit OR zone) near3 address$ WITH eras$ near5 (page OR block)	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:23 PM
L29	121	L26 AND flash WITH controller AND interface WITH controller AND (map$ OR translat$) WITH physical WITH logical AND (translat$ OR map$) WITH (block OR page)	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:23 PM
L30	69	L29 AND (query OR request) near3 respon$	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:24 PM
L31	20	L27 AND (query OR request) near3 respon$	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/21 06:24 PM
L32	140	(command OR request OR query) near3 respon$ AND (maintain$ OR preserv$) WITH (unit	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB,	OR	ON	ON	2022/12/21 06:28 PM

		OR zone OR block OR page) near3 address$ WITH eras$ near5 (page OR block)	HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L33	1	L32 AND (translat$ OR map$) WITH page near3 address WITH (eras$ OR invalidat$) WITH table	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/21 06:28 PM
L34	25	L32 AND (translat$ OR map$) WITH page near3 address AND (eras$ OR invalidat$) WITH table	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/21 06:28 PM
L35	703	(command OR request OR query) near3 respon$ AND (maintain$ OR preserv$) WITH (unit OR zone OR block OR page) near3 address$ WITH eras$ near5 (page OR block)	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:29 PM
L36	2	L35 AND (translat$ OR map$) WITH page near3 address WITH (eras$ OR invalidat$) WITH table	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:29 PM
L37	54	L35 AND (translat$ OR map$) WITH page	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:30 PM

		near3 address AND (eras$ OR invalidat$) WITH table					
L38	16	(lookup OR look?up) near3 table WITH eras$ near5 address$2 AND (preserv$ OR reserv$ OR sav$3) WITH address$	(USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/21 06:32 PM
L39	45	(lookup OR look?up) near3 table WITH eras$ near5 address$2 AND (preserv$ OR reserv$ OR sav$3) WITH address$	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:33 PM
L40	0	flash WITH controller AND interface WITH controller AND (identif$ OR translat$) WITH physical WITH logical AND (translat$ OR map$) WITH pahe near3 address WITH (eras$ OR invalidat$) WITH table	(USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/21 06:34 PM
L41	0	flash WITH controller AND interface WITH controller AND (identif$ OR translat$) WITH physical WITH logical AND (translat$ OR map$) WITH pahe near3 address WITH (eras$ OR invalidat$) WITH table	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:35 PM
L42	13	flash WITH controller AND (maintain$ OR preserv$) WITH (block OR page) near3 address$ WITH (eras$ OR invalidat$) AND (map$ OR translat$)	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:36 PM

		WITH physical WITH logical AND (translat$ OR map$) WITH page near3 address WITH (eras$ OR invalidat$) WITH table					
L43	13	flash WITH controller AND (maintain$ OR preserv$) WITH (block OR page) near3 address$ WITH (eras$ OR invalidat$) AND (map$ OR translat$) WITH physical WITH logical AND (translat$ OR map$) WITH page near3 address WITH (eras$ OR invalidat$) WITH table	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/21 06:36 PM
L44	20	(empty OR full) near3 (state OR status OR flag) AND (maintain$ OR preserv$) WITH (block OR page) near3 address$ WITH eras$ near5 (page OR block)	(USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/21 06:38 PM
L45	83	(empty OR full) near3 (state OR status OR flag) AND (maintain$ OR preserv$) WITH (block OR page) near3 address$ WITH eras$ near5 (page OR block)	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:38 PM
L46	0	L45 AND (translat$ OR map$) WITH page near3 address WITH (eras$ OR invalidat$) WITH table	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:39 PM
L47	277	(maintain$ OR preserv$) WITH (block OR page) near3 address$ WITH eras$ near5 (page OR block)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL,	OR	ON	ON	2022/12/21 06:39 PM

			NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L48	21	L47 AND (translat$ OR map$) WITH page near3 address WITH (eras$ OR invalidat$) WITH table	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/21 06:40 PM
L49	780	(maintain$ OR preserv$) WITH (block OR page) near3 address$ WITH eras$ near5 (page OR block)	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:40 PM
L50	10	L49 AND (translat$ OR map$) WITH page near3 address WITH (eras$ OR invalidat$) WITH table	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:41 PM
L51	9	address near3 (range OR block) WITH (host OR processor) WITH (permi$ OR allow$) near3 access$2 AND logical near3 address$ AND (host OR processor) near5 (accessible OR addressable) near3 (block OR page OR range OR zone)	(USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	OR	ON	ON	2022/12/21 07:06 PM
L52	19	address near3 (range OR block) WITH (host OR processor) WITH (permi$ OR allow$) near3 access$2 AND logical near3 address$ AND (host OR processor) near5 (accessible OR	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 07:06 PM

		addressable) near3 (block OR page OR range OR zone)					

PE2E SEARCH - Search History (Interference)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	British Equivalents	Time Stamp
N1	0	((lookup OR look?up) near3 table WITH eras$ near5 address$2 AND page near2 address$ WITH (preserv$ OR reserv$ OR sav$3)).clm.	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/17 11:04 AM
N2	0	((lookup OR look?up) near3 table WITH eras$ near5 address$2 AND address$ WITH (preserv$ OR reserv$ OR sav$3)).clm.	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/17 11:05 AM
N3	16	((command OR request OR query) near3 respon$ AND (maintain$ OR preserv$) WITH (unit OR zone OR block OR page) near3 address$ WITH eras$ near5 (page OR block)).clm.	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:27 PM
N4	0	((lookup OR look?up) near3 table WITH eras$ near5 address$2 AND (preserv$ OR reserv$ OR sav$3) WITH address$).clm.	(US-PGPUB; USPAT)	OR	ON	ON	2022/12/21 06:33 PM

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 157 328 B1

(12) **EUROPEAN PATENT SPECIFICATION**

(45) Date of publication and mention of the grant of the patent:
04.05.2005 Bulletin 2005/18

(21) Application number: **00903861.3**

(22) Date of filing: **17.02.2000**

(51) Int Cl.[7]: **G06F 3/06**

(86) International application number:
PCT/GB2000/000550

(87) International publication number:
WO 2000/049488 (24.08.2000 Gazette 2000/34)

(54) **MEMORY SYSTEM**

SPEICHERSYSTEM

SYSTEME DE MEMOIRE

(84) Designated Contracting States:
DE FR GB

(30) Priority: **17.02.1999 GB 9903490**

(43) Date of publication of application:
28.11.2001 Bulletin 2001/48

(73) Proprietor: **LEXAR MEDIA, INC.**
Fremont, CA 94538 (US)

(72) Inventors:
- **SINCLAIR, Alan, Welsh**
  Del Mar, CA 92014 (US)
- **OUSPENSKAIA, Natalia Victorovna**
  St. Petersburg, 194100 (RU)
- **TAYLOR, Richard, Michael**
  Dalkeith, Midlothian EH22 2AQ (GB)
- **GOROBETS, Sergey, Anatolievich**
  Edinburgh EH9 1SE (GB)

(74) Representative:
Beresford, Keith Denis Lewis et al
BERESFORD & Co.
16 High Holborn
London WC1V 6BX (GB)

(56) References cited:
EP-A- 0 522 780
EP-A- 0 712 067
WO-A-94/20906
WO-A-97/37296
GB-A- 2 291 991
US-A- 5 602 987

Note: Within nine months from the publication of the mention of the grant of the European patent, any person may give notice to the European Patent Office of opposition to the European patent granted. Notice of opposition shall be filed in a written reasoned statement. It shall not be deemed to have been filed until the opposition fee has been paid. (Art. 99(1) European Patent Convention).

EP 1 157 328 B1

Printed by Jouve, 75001 PARIS (FR)

Description

[0001] The present invention relates to a solid state memory system for data storage and retrieval, and to a memory controller for controlling access to a non-volatile memory of a solid state memory system. In particular, the invention relates to FLASH memory systems and controllers for FLASH memories.

[0002] FLASH EEPROM (electrically erasable programmable read only memory) devices are commonly used in the electronics industry for non-volatile data storage. Various types of FLASH memory devices exist, including devices based on NAND type memory cells, AND type memory cells, or NOR type memory cells. Such devices may have different types of interfaces to the host processor system(s) for which they are designed to interface, for example they may use a serial access type interface (as commonly used in many NAND and AND type devices) or a random access type interface (as used in some NOR type devices). The present invention is intended to be applicable, in appropriate forms, to at least some and preferably all of these different types of memory devices.

[0003] It is known to use solid state memory systems to try to emulate magnetic disc storage devices in computer systems. It is an aim of the industry to try to increase the speed of operation of solid state memory systems so as to better emulate magnetic disc storage.

[0004] It is known from GB-A-2291991 to provide a flash memory with address translation tables stored in a secondary memory to allow translation of logical to physical addresses.

[0005] It is also known from EP-A-0712067 for a conversion table to be stored in the flash memory itself. Spare sectors are provided in each block of memory to allow updating of memory blocks without having to update the conversion table each time a sector content is updated.

[0006] According to a first aspect of the present invention we provide a memory system for connection to a host processor, as defined in claim 1.

[0007] Embodiments of the invention are set out in the dependent claims.

[0008] By not updating the SAT every time data from the host processor is written to a sector in the memory, but instead updating the SAT on a less frequent basis, the invention provides very high speed operation of solid state memory, for example FLASH memory, thereby enabling good emulation of magnetic disk memory.

[0009] The physical sector addresses in the SAT are preferably ordered by logical sector address, whereby the Nth SAT entry contains the physical address of a sector to which data having logical address N has been written. When a sector read command is received from the host processor, the controller may look up a logical sector address received from the host processor in the SAT in order to obtain the physical sector address which the controller previously allocated to said logical sector address. The SAT is preferably stored in one or more of said blocks of memory sectors in the solid state memory, each block which contains any portion of the SAT hereinafter being referred to as a SAT block. Preferably the SAT is updated by rewriting one or more blocks of the SAT. By updating a whole block of SAT sectors at a time this significantly speeds up operation of the memory system.

[0010] There may be provided at least one block of sectors (hereinafter referred to as the Additional SAT Block (ASB)), containing modified versions of individual sectors of a said SAT block. Each sector in a said ASB block preferably contains the physical address of the sector of the SAT block which it updates, and the modified version of the said SAT sector. The purpose of an ASB is to cache individually in solid state memory modified sectors of the SAT so as to reduce the number of SAT block rewrites. When all the sectors in a said ASB block are written to with modified versions of SAT sector (s), the respective SAT block is rewritten so as to include all the modified versions in the ASB block and the ASB block is erased.

[0011] It will be appreciated that in the memory system of the invention the physical address which is allocated to any given logical address received from the host processor is not dependent on the logical address itself. The controller merely allocates the physical sector address to which the Write Pointer is currently pointing.

[0012] As described above, the controller fills one said block which is treated as erased before moving the Write Pointer (WP) on to another block. The controller may conveniently be configured to move the Write Pointer (WP) in a predetermined order through the blocks which are treated as erased.

[0013] The controller may conveniently control the Write Pointer (WP) so as to move sequentially, in ascending numerical order of physical address, through the erased blocks, as each block is filled with data written thereto. The control of the Write Pointer (WP) may be cyclic in the sense that once the sectors in the highest block, according to physical address order, have been filled with data the WP is controlled by the controller to wrap around to the block of sectors having the numerically lowest physical addresses out of all the blocks currently being treated by the controller as erased.

[0014] The controller may, alternatively, use another predetermined order for writing data to the memory sectors. For example, the controller may control the Write Pointer (WP) to move sequentially in descending numerical order, according to physical address, through the blocks which are treated as erased. Another possibility would be to move in non-sequential order through the physical sector addresses. For example, the WP may move in descending numerical address order through the physical sector addresses in each block which is treated as erased, and move from block to block in some predetermined order such as, for example, in ascending numerical order according to the physical ad-

dress of the first sector in each said block.

[0015] It will be appreciated that many other predetermined orders are possible for writing data to the sectors in the blocks which are treated as erased. Furthermore, the controller could use the erased blocks in any other order which need not be predetermined, or which may be only partially predetermined. Although generally not preferred, the erased blocks could even be used in a random order.

[0016] The memory sectors in each said block of sectors are preferably erasable together as a unit. The sectors may also be individually erasable (for example where the solid state memory is AND type memory). The controller is preferably configured to control erase operations on the memory so as to only erase whole blocks of memory sectors. A block of sectors will be treated by the controller as an erased block if all the memory sectors therein are erased sectors. If a block contains one or more bad (i.e. defective) sectors, the controller may define the whole block as being bad and treat that block as a not erased block, whereby no data will be written thereto. Alternatively, if a block contains one or more bad sectors the controller may treat that block as an erased block whereby the controller may still use good sectors in the block to store data. In the latter case, though, the memory system preferably includes a table identifying bad sectors and the controller is configured to check whether the next sector address to which the Write Pointer (WP) is to be moved is the address of a bad sector and, if it is the address of a bad sector, to control the Write Pointer to skip this bad sector and move to the next sector address according to the predetermined order in which the sectors are to be written to.

[0017] Any block which contains any good (i.e. not defective) sectors which have already been written to will be treated by the controller as a not erased block.

[0018] Furthermore, it is intended that the term "erased" sector covers not only a sector which has been erased, but also covers a sector which has never yet been written to, and so has not yet ever been erased. Thus, a block of sectors which have never yet been written to is treated by the controller as an erased block.

[0019] Each block of sectors preferably has a physical block address defining its physical position in the memory. The physical address of each said memory sector will preferably include the physical block address of the block in which it is located.

The controller may advantageously be configured to compile a list of the physical block addresses of at least some of the blocks of sectors being treated as erased, which may be used by the controller in order to quickly identify the next block of sectors to be written to. This list of addresses of erased blocks is preferably stored by the controller in a temporary memory which may be provided in the memory system, which temporary memory may conveniently be an SRAM in a microprocessor of the controller, and may be created from information already stored in the solid state memory by the controller identifying the erased state of each block of sectors. (This information will preferably be held in the form of a bitmap in the solid state memory, in which each block is recorded as an erased block or a not erased block.)

[0020] The controller is conveniently configured so that, when a sector write command is received by the controller from the host processor which command renders obsolete data previously written to another sector, the controller stores in a temporary memory the address of the sector containing the now obsolete data. This temporary memory may conveniently be SRAM or DRAM provided in a microprocessor of the controller. If a sector delete command, generated by a user, is received from the host processor by the controller, the controller preferably marks as obsolete the sector to be deleted (without physically erasing the sector). The controller may allow only one block at any time, hereinafter referred to as the Current Obsolete Block (COB), to contain one or more sectors containing obsolete data which was written by the Write Pointer(WP), and when all the sectors in the COB contain obsolete data, the COB is immediately erased. This is a particularly suitable scheme for the case where the Write Pointer (WP) moves sequentially through the memory sector addresses in each block which is treated as erased before moving on to the next block. In such a scheme, a series of obsolete sectors to be deleted (which may, for example, contain part of a user data file which has been rewritten) will in most cases all be in the same block. When a series of sectors are rewritten in a different order to that in which they were previously written, this may create obsolete sectors in more than one block. Where a sector in a block other than the COB is to contain obsolete data, the controller preferably relocates any data in valid (not obsolete) sectors in the COB to another block, which may be the block to which the Write Pointer (WP) is currently pointing, and then erases the COB. Said sector in the block other than the COB is then marked as obsolete and this other block is now the COB. Rather than writing the relocated data to the current location of the Write Pointer, the memory system may include a second write pointer, hereinafter referred to as the Relocation Pointer (RP), for pointing to the physical address of the sector to which such relocated data is to be written, the Relocation Pointer (RP) always being in a different block of sectors to the Write Pointer (WP). This has the advantage of preventing relocated data from being intermingled with data structures directly ordered to be written by the host processor i.e. written by the Write Pointer (WP).

[0021] Generally, only two types of data are written to the solid state memory from the host processor. These are file data and system data. To further reduce the number of reallocations and erasures, the memory system may further include a third write pointer, hereinafter referred to as the System Write Pointer (SWP), which points to the physical address of the sector to which sys-

12/21/2022 19:59:42 Page 3 of 57
SAMSUNG EX1002, Page 217 of 1031

tem data is to be written from the host, the SWP always being in a different block to the Write Pointer (WP) (and in a different block to the Relocation Pointer, if there is one). System data will preferably be identified during initialisation of the system and will be updated as necessary during operation.

[0022] Where both a write pointer (WP) and a system write pointer (SWP) are provided, file data will in this case always be written to the addresses pointed to by the Write Pointer (WP). Both the Relocation Pointer (RP) and System Write Pointer (SWP) are preferably controlled to move through the physical addresses of the memory sectors in said blocks which are treated as erased in a similar manner to the Write Pointer (WP). Thus, when all the (good) sectors in a said block have been filled with relocated data or system data, the respective one of the Relocation Pointer (RP) and the System Write Pointer (SWP) moves on to the next address defined by the controller to be used from the physical addresses of all the sectors in the blocks treated as erased.

[0023] Where a System Write Pointer (SWP) is provided, the controller will preferably allow at least two blocks which contain one or more obsolete sectors to exist at any time, one being said COB and the other being a Current Obsolete System Block (COSB) containing one or more obsolete system data sectors. If any system data sectors need to be relocated in order to allow the COSB to be erased, the relocated system data is preferably sent to the address to which the System Write Pointer (SWP) is currently pointing.

[0024] In fact, there may temporarily exist more than two blocks (the COB and COSB) containing obsolete data at any one time. It is possible that when the COB, for example, needs to be erased (obsolete data has just been created in another block) one of the write pointers may be pointing thereto i.e. the WP is still writing to the block which is currently the COB. Where this is the case the controller preferably proceeds with creating the new COB but postpones the erasure of the old COB (which is hereinafter treated as the Pending Obsolete Block (POB)) until all erased sectors in the POB have been filled and the write pointer moves on to the next erased block to be used, as defined by the controller. At this time any valid (not obsolete) data in the POB is relocated and the POB is erased.

[0025] In addition to writing data structures to the memory from the host processor, the controller may also generate and write to the memory data designated as control information. The controller preferably writes such control information in separate ones of the blocks of memory sectors to those in which data structures received from the host processor are written. Blocks for storing such control information, hereinafter referred to as Control Blocks (CBs), will be updated periodically by the controller and will be accessed during initialisation, and occasionally during operation, of the memory system.

[0026] The controller preferably stores in a temporary memory (which may be a RAM provided in the memory system or which may conveniently be an embedded SRAM or DRAM in a microprocessor of the controller) a list of logical sector addresses for data structures which have been written by the Write Pointer (WP) since the SAT was last updated. This list stored in the SRAM is hereinafter referred to as the logical addresses in the WSL are advantageously stored in the order in which they were written to the non-volatile sectors in the memory. Conveniently, for a group of consecutively written sectors, the WSL entry may therefore be written as the first sector logical address and the sector group length i.e. the number of sectors written. Each said sector group is defined so as not to span more than one block of sectors.

[0027] The controller advantageously also stores in said temporary memory the order in which blocks have been used by the Write Pointer (WP) for writing data since the last update of the SAT. This is stored in the form of a list of block addresses of the blocks in which the updated sectors whose addresses are held in the WSL are located. This list of block addresses is hereinafter referred to as the Write Block List (WBL). It will be appreciated that since the memory system, by virtue of the WSL and WBL, contains knowledge of the location in physical memory which was allocated for the first logical address in said group of consecutively written sectors, the controller can thus always access the correct physical sector for each logical sector address in a said group of consecutively written sectors written since the last SAT update, using the WSL and WBL. The WSL will preferably have a predetermined size and once the WSL is full one or more SAT blocks (and/or ASBs) may be updated and the WSL and WBL are emptied.

[0028] Preferably, the starting physical sector address, and the links between blocks containing sectors to which data has been written by the controller since the last SAT update, are also stored in a Control Block of the solid state memory. By storing the logical sector address for the user data stored in each sector in the sector itself, for example in a header field provided in the sector, the WSL and WBL can therefore easily be recreated following any removal and restoration of power to the system by scanning through the solid state memory, reading the logical addresses in the sectors written to since the last update of the SAT, until reaching a block which is not full. This is the block which contained the Write Pointer (WP) before removal or loss of power. This provides high data security in the event of unexpected power removal from the memory system.

[0029] Where a Relocation Pointer and a System Write Pointer are included in the memory system, the controller preferably also stores in said temporary memory (e.g. SRAM or DRAM in the controller microprocessor) similar lists of logical sector addresses corresponding to sectors in the memory to which relocated data or system data has been written to respectively, which lists

are hereinafter referred to as the Relocation Sector List (RSL) and Write System Sector List (WSSL) respectively. The controller may also store in said temporary memory corresponding lists of the order of blocks which have been used by the RP and the SWP, similar to the Write Block List, and these two lists will hereinafter be referred to as the Relocation Block List (RBL) and the Write System Block List (WSBL). Moreover, the starting physical sector address, and the links between blocks containing sectors to which relocated data or system data has been written since the last SAT update may also be stored in at least one said Control Block (CBs) of the solid state memory whereby the RSL and WSSL can be recreated following any removal and restoration of power to the host processor by simply scanning the memory and reading the logical addresses in the sectors written to by the RP and SWP respectively, since the last update of the SAT.

[0030] Each said sector in any of the above-described embodiments may consist of a single "page" of memory i.e. one row of memory cells in a said block of memory sectors. However the invention is not limited exclusively to such a sector format and in some cases (for example when using random access NOR type memory) each said sector may be less than, or greater than, one page. Moreover, in the latter case not all said sectors need necessarily be of the same size. For example, a data organisation scheme such as that described in our earlier International Patent Application No.PCT/GB99/00188 could be used by the controller to form sectors of appropriate sizes so as to avoid individual defects (of sub-sector size) which may be present in the solid state memory.

[0031] Each sector is, as aforesaid, individually addressable. Each sector may comprise a plurality of sector portions which are also each individually addressable and the controller may write to, and read from, each sector portion individually. It will be appreciated that the smallest possible sector portion size is the minimum addressable unit of the memory. In NOR type memory, for example, the minimum addressable unit of memory is commonly 1 byte.

[0032] The controller preferably writes data to, and reads data from, the memory sectors in uniformly sized data segments. Where all the memory sectors are the same size, each said data segment is preferably equal in size to the size of a said memory sector. Each data segment may comprise data structures from the host processor (e.g. file or system data) and/or data generated by the controller.

[0033] Where the solid state memory is based on NAND type devices, the controller preferably stores in said one or more Control Blocks a list of the block addresses of blocks in the non-volatile memory containing bad sectors (hereinafter referred to as the Bad Block List (BBL)), and the controller treats each such block as a "not erased" block, so that it will not appear in the list of erased blocks which may be stored in temporary memory, and the controller will not write any data to that block.

[0034] Where the memory is based on AND type devices, the controller preferably stores in said one or more Control Blocks (CBs) a list of addresses of any bad sectors, and the controller controls the said write pointer(s) to use the good sectors in any block containing at least one bad sector, and to skip any bad sectors. It will be appreciated that in the latter case where a block containing one or more bad sectors is to be erased the good (i.e. non-defective) sectors in the block are erased individually during a block erase operation.

[0035] The controller advantageously also stores in said one or more Control Blocks a list of the block addresses of all SAT blocks. This list is preferably in the form of a plurality of list portions, each said portion being hereinafter referred to as a Table Block List (TBL), and each said portion containing the block addresses of a group of logically contiguous SAT blocks and any corresponding ASBs.

[0036] The controller preferably stores the block addresses of said one or more Control Blocks in a dedicated block of the memory hereinafter referred to as the Boot Block (BB). Other important information required for data security may also be stored in the Boot Block, for example the list of bad blocks (or bad sectors). Preferably, the first block of sectors in the memory which does not contain any bad sectors is designated as the Boot Block (BB).

[0037] Preferably, the controller will only use blocks containing all good sectors as SAT blocks, Control Blocks, ASBs or BBs.

[0038] A cache may be provided in temporary memory (for example RAM in the memory system, such as SRAM or DRAM in the controller microprocessor), in which the controller stores a group of contiguous SAT entries including the SAT entry most recently accessed from the SAT (by the controller). This further improves address translation speed. Further increase in speed of address translation may be achieved by creating in said temporary memory a list of physical addresses of all ASBs and the SAT blocks with which they are associated (hereinafter referred to as the ASB List or ASBL) which is updated each time a SAT sector write operation is performed. Similarly, the positions of the TBLs in the Control Block(s) may also be stored in said temporary memory so as to allow even faster logical-to-physical sector address translation.

[0039] The solid state memory may comprise a single memory array in the form of a single memory chip, or may comprise a plurality of memory arrays in the form of a plurality of memory chips. Where the memory comprises a plurality of chips, the controller advantageously forms the memory sectors in the plurality of memory chips into a multiplicity of virtual blocks, each said virtual block comprising one erasable block of memory sectors from each said memory chip, and the controller preferably sorts said virtual blocks into ones which are treated

12/21/2022 19:59:42 Page 5 of 57
SAMSUNG EX1002, Page 219 of 1031

as erased and ones which are treated as not erased. The controller preferably compiles a list of the virtual blocks treated as erased and stores this in temporary memory in the memory system, which may be SRAM in a microprocessor of the controller. The controller preferably controls the Write Pointer (WP) (and the RP and SWP, where provided) to move from one chip to another for each consecutive sector write operation, starting at one sector in one erasable block of the virtual block and moving consecutively to one sector in each of the other erasable blocks in the virtual block until one sector has been written in each erasable block of the virtual block, and then moving back to the chip in which the first sector was written and proceeding in a similar manner to fill another one sector in each erasable block of the virtual block, and so on until the virtual block is full of data. The Write Pointer (WP) then moves on to the next virtual block in said list of virtual blocks being treated as erased, and fills this next virtual block in a similar manner. The controller is preferably configured so that for every n contiguous sector write operations the controller executes, where n is less than or equal to the number of solid state memory chips in the memory system, the controller writes substantially concurrently to one sector in each of n of the chips. The controller preferably carries out erasure of any said virtual block by concurrently erasing all the erasable blocks in the virtual block.

[0040] It will be appreciated that the controller of the memory system may be substantially implemented in circuitry as a controller device, but will preferably be implemented, at least in part, as firmware held in the memory of a controller device. The controller may be integrally formed on the same chip (or one of the same chips) as the solid state memory.

[0041] According to an embodiment of the invention we provide a memory system according to claim 1, the memory system further comprising:

a solid state memory comprising a plurality of solid state memory chips each having non-volatile memory sectors which are individually addressable and which are arranged in erasable blocks of sectors, each said sector having a physical address defining its physical position in the memory;
and a controller for writing data structures to and reading data structures from the memory, wherein:

the controller forms the erasable blocks into virtual blocks, each said virtual block comprising an erasable block from each of the memory chips, and the controller sorts the virtual blocks into ones which are treated as erased and ones which are treated as not erased, and the controller fills one virtual block with data prior to moving on to the next virtual block to be filled, and each virtual block is filled by writing to the memory sectors thereof in a repeating sequence in which the controller writes to one memory sector in each of the erasable blocks of the virtual block one after another whereby consecutively written sectors are in different chips.

[0042] Preferably, the controller is configured so that for every n contiguous sector write operations the controller executes for a multiple sector write command received from the host processor, where n is less than or equal to the number of solid state memory chips in the memory system, the controller writes substantially concurrently to one sector in each of the n of the chips.

[0043] According to a second embodiment of the invention we provide a controller for writing data structures to and reading data structures from a solid state memory having non-volatile memory sectors which are individually addressable and which are arranged in erasable blocks of sectors, each said sector having a physical address defining its physical position in the memory, wherein the controller includes:

means for translating logical addresses received from a host processor of a memory system in which the controller is used to physical addresses of said memory sectors in the memory, and for sorting the blocks of sectors into blocks which are treated as erased and blocks which are treated as not erased;
and a Write Pointer (WP) for pointing to the physical address of a sector to which is to be written to from the host processor, said Write Pointer (WP) being controlled by the controller to move in a predetermined order through the physical addresses of the memory sectors in any block which is treated as erased and, when the block has been filled, to move to another of the erased blocks;
and wherein, when a sector write command is received by the controller from the host processor, the controller translates a logical sector address received from the host processor to a physical address to which data is written by allocating for said logical address that physical address to which said Write Pointer (WP) is currently pointing;
and wherein the controller is configured to compile a table (the SAT) of logical addresses with respective physical addresses which have been allocated therefor by the controller, and to update the SAT less frequently than memory sectors are written to with data from the host processor.

[0044] According to a second aspect of the invention we provide a method of controlling reading and writing of data structures to and from a solid state memory having non-volatile memory sectors which are individually addressable and which are arranged in erasable blocks of sectors, each said sector having a physical address defining its physical position in the memory, the method being as defined in claim 50.

[0045] Preferred embodiments of the invention will

12/21/2022 19:59:42 Page 6 of 57
SAMSUNG EX1002, Page 220 of 1031

now be described by way of example only and with reference to the accompanying drawings in which:

Fig.1 is a schematic illustration of one block of sectors in a NAND type FLASH memory, showing three sectors therein;
Fig.2 is a block diagram of a memory system comprising a FLASH chip and a controller chip;
Fig.3 is a schematic illustration of one page of data in a NAND or AND type FLASH memory;
Fig.4 shows the structure of a Header field of the page of Fig.3;
Fig.5 illustrates the format of a physical address (PA) of a page;
Fig.6 illustrates a Control Block (CB) entry;
Fig.7 illustrates one entry in a Table Block List (TBL);
Fig.8 shows the format of a MAP entry;
Fig.9 shows the format of an entry in the ASB List (ASBL);
Fig.10 illustrates the format of a Current Obsolete Block (COB) Structure;
Fig.11 is a table illustrating the order in which sectors are written to in a virtual block of a multiple FLASH chip memory system according to one embodiment of the invention;
Fig.12 shows the format of a Virtual Address (VA);
Fig.13 shows how the PA is obtained from the VA;
Fig.14 illustrates the timing of operations during a multiple sector write to a multiple FLASH chip memory system according to the invention;
Fig.15 is a block diagram of a controller chip;
Fig.16 is a table showing allocated memory capacity for a memory system of the invention;
Fig.17 is a flow diagram showing an Address Translation process;
Fig.18 is a flow diagram of the steps carried out at box 58 of Fig.17;
Fig.19 is a block diagram of a multiple FLASH chip memory system comprising four FLASH chips and a controller chip;
Fig.20 is a flow diagram of the steps carried out at box 56 of Fig.17;
Fig.21 is a flow diagram of the steps carried out at box 44 of Fig.17;
Fig.22 is a flow diagram of a sector read operation;
Fig.23 is a flow diagram of a sector write operation;
Fig.24 is a flow diagram of the steps carried out at box 161 of Fig.23;
Fig.25 is a flow diagram of the steps carried out at. box 207 of Fig.24;
Fig.26 is a flow diagram of the steps carried out at box 160 227 of Fig.23;
Fig.27 is a flow diagram of a sector delete operation;
Fig.28 illustrates the physical partitioning of a page in NAND or AND type FLASH memory;;
Fig.29 is an illustration of an alternative way of arranging the data in the FLASH page of Fig.28;
Fig.30 is an illustration of a yet further way of arranging the data in the FLASH page of Fig.28;
Figs.31(a) illustrates data in a buffer memory of the controller prior to a sector write operation;
Fig.31(b) illustrates data in a FLASH page after completion of a write operation, where the data is arranged according to the embodiment of Fig.30;
Fig.32 is a table of controller commands used to transfer the data from the controller buffer to the FLASH memory during the write operation of Figs. 31(a) and (b);
Fig.33 illustrates data in a buffer memory of the controller after a read operation;
Fig.34 is a table of of controller commands used to transfer the data from the FLASH memory to the controller buffer during the read operation of Fig.33; and
Fig.35 is a schematic block diagram of an erasable block of sectors in NOR type FLASH memory, showing three sectors therein.

[0046] Fig.1 illustrates schematically the physical page structure in one block 4 of a FLASH memory array based on NAND type memory cells. Fig. 1 shows three pages 1, 2, 3 in the block 4. The page 1 in physical terms comprises one row of memory cells in a block of memory, the memory being partitioned into many such blocks each comprising a multiplicity of rows of memory cells (i.e. a multiplicity of pages). Each page 1, 2, 3 is treated as one sector of physical memory space in the FLASH memory system which will be described and is 528 Bytes wide. Each page 1 in the memory is individually addressable (for read/write and delete operations), and the pages are erasable in blocks. We will now describe a memory system incorporating such a memory array. We will later additionally describe memory systems based on AND or NOR type FLASH memory.

[0047] Fig.2 shows a memory system 10 incorporating a FLASH memory chip 5 and a controller chip 8. The FLASH memory chip 5 comprises a FLASH memory array 6 and a read/write buffer 7 interfaced to a controller buffer 9 in the controller chip 8. The controller chip 8 further includes a controller microprocessor 11 and an Error Correction Code (ECC) generator and checker 12. The controller buffer 9 interfaces to a host computer processor (not shown) connected to the memory system 10 via an output O/P of the controller chip 8. The controller chip 8 (hereinafter referred to as the "controller"), controls the reading and writing of data structures to and from the memory array 6. The host processor 2 connected to the memory system 10 sends read and write commands to the controller 8. Data can be accessed by the host in 512 Byte portions or "host data sectors", each of which has a logical sector address (LA). The controller 8 receives an LA from the host processor and translates this to a physical address as will be described hereinbelow. In the present case (NAND type memory), each physical address (PA) defines the physical position of a

12/21/2022 19:59:42 Page 7 of 57 SAMSUNG EX1002, Page 221 of 1031

page 1 of the FLASH memory in the array 6. Each LA is in the form of one 24 bit field. Accessing a PA using an LA is referred to as address translation and is commonly the most frequent operation needed on every read/write access. The controller 8 writes data to the memory array 6 in data segments, each segment being 528 Bytes wide. For each 512 Bytes of data received from the host (e.g. user file or system data) the controller generates 16 Bytes of data comprising a 4 Byte Header generated by the microprocessor 11 and a 12 Byte ECC produced by the ECC generator and checker 12. The controller organises this into a 528 Byte data segment which is written to one page of the memory array 6, via the FLASH buffer 7. The logical address (LA) of a host data sector is stored in the 4 Byte Header in the FLASH Sector 1 in which that host data sector is written. On a read operation the data stored in the relevant sector of the FLASH memory array is read from array 6, via the FLASH read/write buffer 7, into the controller buffer 9 (and concurrently to the ECC generator and checker to check for errors in the data), and the controller reads the 4-byte header to check that the stored LA matches the LA requested by the host computer, prior to allowing the host computer to read the data from the controller buffer 9.

[0048] The Controller 8 manages the physical location of data written to the memory 6 on an individual sector basis. As will be later described in further detail, the controller stores in the memory 6 a BitMap (MAP) of erased blocks and compiles in SRAM in the microprocessor 11 a list (the Next Erased Block (NEB) list) of at least some erased blocks, ordered in ascending order of block physical addresses, which erased blocks are to be used for writing to. The physical page location at which a host data sector is written does not depend on the logical address received from the host. Each Host Data sector is written at an address defined by a cyclic write pointer. Special write pointers are used for different types of write operations; host file data writes are performed at an address pointed to by the Data Write Pointer (WP), host system data writes at an address pointed to by the System Write Pointer (SWP). The Relocation Pointer (RP) is used for writing sectors which were not directly ordered by a host. Each of these write pointers has an. identical behaviour: each pointer moves sequentially through the pages of a block, then moves to the first page of the next erased block in the Next Erased Block (NEB) list. Blocks containing files which are not erased are treated as "not erased" blocks and are skipped when a pointer moves from one block to another (and are never included in the NEB).

Sector Relocation Algorithm

[0049] When a block 4 of sectors is to be erased, so as to recover sector space containing obsolete data, sectors must be relocated from a block containing a combination of valid and obsolete sectors to allow the block to be erased. In principle, the controller 8 allows only one block corresponding to a specific write pointer to contain obsolete data sectors at any time. When a sector to be written by the host would produce an obsolete sector in a second block, the existing block must first be erased, after relocation of valid sectors, if necessary.

[0050] Therefore, numerous erasures and extensive relocations of sectors are unavoidable when a majority of blocks are to contain both valid and obsolete sectors. This occurs only when a sequence of sectors written by a host as a part of a file differs from the sequence in which they had previously been written. This is not the normal case in most applications. However, even in case of normal file write operation, relocation of non-related data must be performed from blocks containing a "head" and "tail" of a file. There is a high probability that relocation of system data intermingled with "other file" data will cause additional erasure of another block and this will produce more relocations from this block.

[0051] To reduce the total number of relocations and erasures system data is therefore specifically identified and is always written or relocated at the address of the System Write Pointer (SWP). Information about system data is obtained during initialisation process and is stored in the microprocessor SRAM 13. It will be generally understood that a data file is written to the FLASH memory by a file system in the host computer processor. File data is partitioned into clusters by the file system, where each cluster is a group of contiguous host data sectors of (typically) 512Bytes. The file system maintains tables and data structures relating to the attributes of the files stored in the memory and the locations of the clusters which form each file. These tables and structures are stored in the FLASH memory (as system data) and are updated by the file system whenever file data is written. When file data is written to the memory it is accompanied by system data (e.g. for Directory and File Allocation Tables) which relate to the file. System data written in the memory commonly includes BIOS Parameter configuration information, one or two copies of the File Allocation Table (FAT) in which each entry relates to a specific cluster, the Root Directory, and Subdirectory information. The controller is configured to recognise an operation to write a host system data sector, thereby enabling it to treat this host data sector differently to a host data sector of file data. A number of methods may be used either singly or together to recognise system sector writes, as follows:

1. System data is written with single sector write commands, whilst file data may be written with multiple sector write commands.
2. All sectors with LAs below the last sector address in the file system Root Directory are system sectors. This address can be determined from information held in a BIOS parameter block stored in the memory by the host file system.

12/21/2022 19:59:42 Page 8 of 57
SAMSUNG EX1002, Page 222 of 1031

3. All sectors within Subdirectories are system sectors. Subdirectory addresses and sizes can be identified by reading all Root Directory and Subdirectory entries.
4. System sectors are often read by a file system immediately before they are rewritten.

[0052] For the same purpose file data sectors to be relocated are written at the address defined by a Relocation Pointer (RP), and are therefore not intermingled with sectors written by the host.

[0053] In a modified embodiment of the invention, an additional write pointer may be provided to point to the location to which relocated system data is to be written. This additional pointer is referred to as the System Relocation Pointer (SRP), and is always located in a different block to the WP and SP.

Block Erase Algorithm

[0054] No arbitrary selection of a block to be erased or scheduling of background erasure is carried out in the present embodiment Listed erasure of a block containing obsolete sectors is normally immediately performed when an obsolete sector in a second block will result from a host sector write command which is pending. Similarly, a block is immediately erased when it contains totally obsolete control data structures as a result of the rewriting of a control block. (Control blocks are where the controller 8 writes certain control data, and are described in further detail later).

[0055] Therefore, normally, since relocations cannot produce obsolete data, there can exist not more than two blocks which contain obsolete data; the Current Obsolete Block (COB) corresponding to the Data Write Pointer (WP) and containing obsolete file data and the Current Obsolete System Block (COSB) corresponding to the System write Pointer (SWP) and containing obsolete system data. However, temporarily there may exist one more obsolete block of each type. This will occur when a block to be erased (obsolete data has just been created in another block) at this moment also contains a write pointer of any type. In this case erasure of such a block (designated Pending Obsolete Block (POB)) has to be postponed until all erased pages in this block have been used and the relevant write pointer will be moved to another block. At this moment the Pending Obsolete Block is immediately erased.

[0056] As aforementioned, erased block identity is maintained in a Bitmap (MAP) spanning the whole FLASH block address space, in which MAP the erased state of each block is recorded. Erased blocks are consumed for sector or control data writing sequentially in block address order. No background erasure is carried out. Any block containing one or more bad sectors is treated as a Bad Block and is treated as a "not erased" block by the controller.

Wear Levelling

[0057] The use of cyclic write pointers and single sector write management produces inherent wear levelling in the FLASH memory. However, the algorithm of erasing blocks as soon as they are populated with obsolete or deleted data produces a wear levelling characteristic which is a function of the sequences of sector write operations. If any further wear levelling is thought to be necessary, then separate additional techniques may be incorporated, such as occasional relocation of sectors in random blocks so as to allow these blocks to be erased.

Address Translation Principles

[0058] The principal address translation means is the Sector Address Table (SAT), which basically is a list of physical addresses of sectors, ordered by logical address. Thus the Nth SAT entry normally contains the physical address for. sector with logical address N. The SAT is organised as a number of independent blocks (SAT blocks), and is updated by rewriting individual pages of the SAT blocks. A SAT block may have a dedicated Additional SAT Block (ASB) linked with it to allow modification of individual pages of the SAT Block. SAT pages are not rewritten after each sector write, but on a much less frequent basis, to minimise impact on sector write performance.

[0059] Therefore, the SAT will not contain the correct physical address for sectors written since the SAT was last updated. The logical addresses of such sectors are stored by the processor in its SRAM 13 in lists called the Write Sector List (WSL), Relocation Sector List (RSL) and Write System Sector List (WSSL). These lists exactly match the ordering of the sectors themselves which were written by a host or relocated from blocks before erasure. In the case of consecutively written sectors the WSL and RSL entry defines first sector logical address and the sector group length. The sector groups cannot jump from one block to another. The microprocessor 11 has knowledge of the starting point of the sector series in Flash memory, and also the order in which blocks were used for sector writing (special lists, complimentary to those described above and created in the processor SRAM, namely the Write Block List (WBL), Write System Block List (WSBL) and Relocation Block List (RBL), are used to store this information and are described in further detail later), and so can calculate the physical location of the sector.

[0060] The WSL, RSL and WSSL (and lists, complimentary to them: WBL, WSBL and RBL) can be recreated by the microprocessor 11 after removal and restoration of power to the memory system 10 by reading the logical addresses in the headers of the sectors in the series written since the last SAT rewrite. The starting sector address in the series and the links between blocks containing sectors in the series is obtained by the

12/21/2022 19:59:42 Page 9 of 57
SAMSUNG EX1002, Page 223 of 1031

microprocessor from entries in a special data structure called the Control Block (CB) in Flash memory. (The Control Block (CB) is described in detail later). This method gives high security of data in the event of unexpected power removal from the card.

[0061] It will be appreciated that where a System Relocation Pointer (SRP) is included, as mentioned above, a System Relocation Sector List (SRSL) and complementary System Relocation Block List (SRBL) are also created and used in a similar manner as described above with regard to the WSL, RSL and WSSL, and the WBL, RBL and WSBL, respectively.

[0062] The organisation of data in a FLASH Sector (i.e. page) 1 to which data is written by the controller 8, according to the present embodiment, is illustrated in Fig. 3. The Sector 1 contains a first, 512 Byte information portion 1a which may, for example, consist of a host data sector, followed by a 4 byte header portion 1b, followed in turn by 12 Bytes of ECC 1c. As shown in Fig. 4, the Header portion itself comprises a Data Structure Type portion 20 and a Header Parameter 22 (e.g. comprising the logical sector address (LA) of a host data sector written in the information portion 1a. The Data Structure Type can have values representing any one of the following : Data Sector; Deleted Data Sector; Sector Address Table (SAT) page; Additional SAT Block (ASB) page; Control Block (CB) page; and Boot Block (BB) page.

Deleted Data Sector

[0063] A deleted data sector physically exists temporarily in the FLASH memory only in blocks which are permitted to contain obsolete or deleted sector data i.e. the COB or COSB. It is identified by the "all zero" state of the Data Structure Type field in the header, or other means as appropriate.

Sector Address Table (SAT)

[0064] The SAT is a series of entries containing the physical address of logical sectors. Entry N contains the physical address for logical sector N. Entries within a page occupy the 512 bytes of the information portion 1a of the data stored in the page. Because a SAT page is written in a single operation, the EEC field may be used to protect the complete page, and separate ECC fields for each entry in the SAT page are not required.

[0065] The SAT is actually stored as a series of blocks, each block containing up to 64 SAT pages. A separate data structure, the Table Block List (TBL), is held in the Control Block (described later) to define the block addresses of all SAT blocks. Each SAT entry defines the physical address of a sector and occupies 3 bytes and as shown in Fig.5 comprises:

Chip Number	5 bits, allows 32 chips to be addressed
Block Number	13 bits, allows there to be 8192 blocks per chip
Sector Number	6 bits, allows up to 64 sectors per block.

[0066] The Header Parameter field on a SAT page data structure contains the SAT block and page number.

[0067] A Flash card with capacity 8MB and 8VB blocks can store approximately 16K sectors, and its SAT will therefore have approximately 16K entries. A 512-byte page within a SAT will contain 170 entries, and the SAT will therefore occupy almost 96 pages, which will occupy 6 blocks. A SAT for a large FLASH memory card (2GB, 8VB blocks) occupies 1543 blocks.

Additional SAT Block (ASB)

[0068] An Additional SAT Block (ASB) is a dedicated block which may be linked to a specific SAT block to allow single pages of the SAT block to be modified (i.e. rewritten). There can be several ASBs, each of which acts as an extension to the SAT block to which it is linked. When a SAT block is to be modified, it generally contains modified data in only a small number of its pages. The ASB allows only these modified pages to be rewritten. This is a much faster operation than the writing of every page and erasure of the obsolete block which is required for rewriting a SAT block. The Header Parameter portion of an ASB page contains the SAT block to which it is linked and the SAT page number within that block which it replaces. The format of the information portion 1a of data stored in an ASB page is identical to that of a SAT page.

Control Blocks (CBs)

[0069] The Controller 8 stores some control and address information in one or more Control Blocks (CBs). The CBs are updated periodically by the controller 8 and must be accessed during initialisation of the memory system 10 or occasionally during operation. Information is stored in the CBs in entries of a fixed size which can be independently written. There are 9 entries per page in each CB. An entry relates to one of the following list of data types, which are identified by a CB Header field in the entry itself :

- Table Block List (TBL)
- Map of blocks with some fields in the entry corresponding to a file data write operation (WMAP)
- Map of blocks corresponding to a system data write operation (SMAP)
- Map of blocks corresponding to a relocate sector operation (RMAP)

When new data must be added to the CB, an additional entry of the appropriate type is added immediately following the last valid entry. For large cards the CB may

12/21/2022 19:59:42 Page 10 of 57
SAMSUNG EX1002, Page 224 of 1031

occupy more than one block. The addresses of all CB blocks are stored in a Boot Block (BB) in the FLASH memory 6.

[0070] For purposes of data security the first page (designated Header Page) of each block of a CB (and the BB also) does not contain entries and has a full page format of like that of Fig. 3. The Header Parameter field of the Header 16 of this page consists of a Signature, which identifies the CB, and its Block Number (which is the block's serial number within the set of Control Blocks).

[0071] The Information Field 1a of the Header Page of the first CB block is occupied by a Block Link Info data. The Block Link Info data provides all necessary information to restore the WBL, WSBL and RBL if the system has to be initialised following a CB rewrite operation. It comprises Link fields collected from all MAPs (WMAPs, SMAPs and RMAPS) written since the last SAT page write operation was performed and has to be written to the Header Page of the first block of a new CB during its rewrite operation. The Block Link Info is a timing ordered list of 4 bytes entries, each of which contains a block address of a block being visited by one of the write pointers and a flag identifying which pointer it was. A maximum number of blocks in memory space is allowed for CBs and when this becomes full, active entries are rewritten to the next available erased block(s) from the NEB (i.e. it is compacted and corresponding Header Pages are added), and the old CBs are erased. (At the same time a corresponding entry has to be added to the Boot Block.) The information field of a CB page contains 9 entries of equal length. The EEC field of a CB page is not used or may be used for other purposes.

[0072] An entry in a CB is 56 Bytes wide and has the format shown in Fig.6 and comprises a Header 24 identifying the data type of the entry, an information field 26, and an EEC field 28. The EEC is of the same form as is used for a full page.

Table Block List (TBL)

[0073] The CBs contain the Table Block List (TBL). The TBL contains the addresses of a group of contiguous blocks of the SAT and any corresponding ASBs. Multiple TBLs are normally required to define all required SAT block addresses. The relevant TBL is written immediately after a SAT block write operation or allocation of a new ASB, to record the modified SAT or ASB block locations. The TBL also contains fields which are used to record write pointer positions at the moment when a SAT page write operation is to be performed. Therefore, a TBL is also written whenever a SAT page write operation is carried out. One entry of the TBL is shown in Fig. 7 and occupies the information field 26 of a CB entry. The first 1 Byte of the TBL entry is No, a sequential number of the TBL entry. Each TBL entry keeps values of 8 SAT - ASB block pairs, in other words each TBL entry keeps values for SAT blocks from N to N+7 where N is No*8.

[0074] WP is the Write Pointer Page field, RP is the Relocation Pointer field, and SWP is the System Write Pointer field. These fields determine the position of the WP, RP and SWP within the blocks after WSL, RSL, SSL release operation. WP, RP and SWP are valid only when WSL, RSL or SSL release operation is complete (i.e. at the last SAT write operation). This condition is set with the Flag bit in the entry Header. Flag=1 means the entry is the last one written during WSL or RSL release and so WP field is valid.

[0075] 3 reserved bytes are left for possible future additions.

[0076] SAT-ASB Pairs follow reserved fields, this is an array of 8 entries, each consists of SAT and ASB block addresses.

[0077] SAT_N is the number of the Nth SAT block and ASB_N is the number of the ASB linked to the SAT_N. If SAT or ASB doesn't exist the value of this field should be equal to zero.

[0078] New TBL entry should be added to CB each time SAT block is relocated or new ASB is linked to SAT block.

MAP (WMAP, SMAP and RMAP)

[0079] The CBs contain various MAP entries. There are three different types of MAP entry each of which corresponds to a different type of write operation; WMAP - to a file data write or delete operation, SMAP - to a system data write and RMAP - to a sector relocation operation.

[0080] The information field of all the MAP entries has the same format. It contains a BitMap defining the erase state of a group of consecutive blocks. The erase state of 256 blocks is defined by the 256 bits in a 32 byte field in the MAP. The information field contains a Range field identifying the group of blocks to which it relates. Another field defines the destination block for a transition of the write pointer between blocks. The MAP also contains fields identifying the location of blocks in flash memory containing obsolete data; ObsC is used for the COB (or COSB) and ObsP - for the Pending Obsolete Block. If obsolete block is not present corresponding field is set to 0. The EB field contains an address of a block in which erasure is caused by current write or delete sector operation. If there is no such block the EB field is set to 0. This MAP entry format is illustrated in Fig.8.

[0081] When one of the write pointers is moved from one block to another, a corresponding MAP entry must be added to show the use of an erased block (the BitMap field is updated) and to record the link between the blocks (the Link field is updated). When a write (or delete sector) operation produces obsolete (or deleted) data in a new block, a corresponding MAP also must be added to record a new obsolete block position (ObsC or/and ObsP fields are updated), to indicate that a block has to

be erased (the EB field is updated) and to show that a new erased block is going to appear (the BitMap field is updated). Therefore, normally, at least two fields of a MAP are written at the same time and this may be achieved in a single page write operation.

Boot Block (BB)

[0082] The function of the Boot Block is to provide high security of data in the event of unexpected power removal from the card and at the same time avoid extensive scanning during initialisation procedure. For large cards the BB contains more than one block. The BB always occupies the first nondefective block(s) in the card. For purpose of data security there is a copy of the BB occupying the next nondefective block(s) in the card. Both the BB and its copy must be put to the same place after being rewritten.

[0083] The BB has the same structure as the Control Block and contains the following types of entries :

- Signature
- Interleaving Enable
- Bad Block List
- Control Block Pointers Table (CBPT)

When new data must be added to the BB, an additional entry of the appropriate type is added immediately following the last valid entry. The Signature and the BBL entries have just the same format as described before (of course, the signature field in the signature entry is different and unique). The Control Block Pointers Table entry contains pointers to all blocks of the CB and has to be updated immediately after the CB is rewritten.

Previous Link (PL)

[0084] The CB(s) also contain the Previous Link (PL). The purpose of the Previous Link is to provide all necessary information to restore the WBL (described later) if the system has to be initialised following a CB rewrite operation. It comprises Link fields collected from all MAPs written since the last SAT page write operation has been performed. The PL has to be written only to a new CB during its rewrite operation.

Data Structures Stored in SRAM of Controller

[0085] Various data structures are stored in the SRAM 13 of microprocessor including the following :

Write Sector List (WSL) (or "whistle)

[0086] The Write Sector List records the logical addresses of sectors written after the last SAT write. Its purpose is to allow correct logical-to-physical address translation for such sectors. The WSL has capacity for 128 entries of length 4 bytes, where each entry stores the logical address of the first sector in a group of consecutively written sectors and the group length. The sector groups cannot jump from one block to another. The WSL is empty immediately after a SAT write.

[0087] The ordering of logical sector addresses in the WSL exactly matches the order in which they were written. A non-volatile copy of the WSL in Flash memory is therefore automatically provided by the headers of the actual sectors written at consecutive locations starting at that defined by the Write Pointer (WP) at the time the SAT was last written. There is therefore no need to explicitly make copies of the WSL to Flash memory 6. If necessary, these sectors can be scanned from a starting address defined in a Control Block field which contains the position of the Write Pointer (WP) at the time the SAT was last written, along with the Link fields from subsequent MAP entries.

[0088] Search of the WSL is performed in reverse order, since only the last entry for any logical sector is valid. Duplicate earlier entries may not have any corresponding obsolete sector located in Flash memory because sector relocations and block erasures may have been performed. If preferred, the controller is configured to simply remove duplicate entries from the WSL.

[0089] Two similar lists, the Relocation Sector List (RSL) and System Sector List (SSL), are also compiled in the microprocessor SRAM 13 recording the logical addresses of relocated sectors and System data sectors (written to addresses pointed to by the RP and SWP respectively) written since the last SAT write.

[0090] An ASB and/or a SAT block is supplemented with WSL, RSL or SSL entries every time the WSL, RLS or SSL respectively is full. This procedure is called WSL, RSL or SSL release. This release can cause ASB release if necessary. ASB release occurs when an ASB is full. When an ASB is full the respective SAT block is rewritten and the ASB is erased. Information about written pages in all ASBs should be stored in RAM to avoid frequent ASB scanning. An ASB List (ASBL) for this purpose is stored in the SRAM 13.

[0091] The ASBL is a list of all the ASB blocks currently linked with SAT blocks, there being one entry in the ASBL for each ASB. Fig.9 illustrates the format of one entry in the ASBL, where :

LWP = number of last written page in this ASB block
NVP = number of the valid pages in ASB block minus one.
ASB Page 0 ... ASB Page n = an array, index is ASB page number, value is corresponding SAT page number.
N = pages per block.

Write Block List (WBL)

[0092] The Write Block List is complementary to the Write Sector List and is created in microprocessor SRAM 13 to define the blocks within which the sectors

12/21/2022 19:59:42 Page 12 of 57 SAMSUNG EX1002, Page 226 of 1031

in the WSL are located. The WBL is empty immediately after WSL release.

[0093] The WSL and WBL are recreated by a scanning process during initialisation of the memory system. The lists in SRAM which are created in this way will exactly match he lists which existed before power was last removed.

[0094] Two similar lists to the WBL named the Relocation Block List (RBL) and System Block List (SBL), are also compiled and stored in the SRAM 13, these lists being complimentary to the RSL and SSL respectively. The RBL and SBL define the blocks within which the sectors in the RSL and SSL respectively are physically located, and are of similar format to the WBL. The RSL, RBL, SSL and SBL can also be recreated by a scanning process during initialisation of the memory system.

Current Obsolete Block (COB)

[0095] Only one block is allowed to exist which contains obsolete or deleted sector data written by the Write Pointer (WP). This is named the Current Obsolete Block (COB). When obsolete or deleted sector data is created in another block, an erase operation must be performed immediately on the block defined as the COB. The current block address of the COB and also a list of sectors which became obsolete or were deleted in this block are stored in the microprocessor RAM 13 as a data structure hereinafter referred to as the COB Structure. The COB Structure is established during initialisation by copying a field containing obsolete block address from the latest MAP entry, and then adding obsolete sector addresses after the reconstructed WSL and WBL have been analysed and reading deleted sector addresses from this block. The COB Structure is updated every time a new delete sector operation is performed or a new obsolete sector is created, and the block address is copied to a current MAP entry in the CB every time obsolete data is created in a new block.

[0096] There is also allowed to exist one block which contains obsolete or deleted sector data written by the SWP, named the Current Obsolete System Block (COSB). The COSB Structure is also stored in the SRAM 13 in a similar manner to the COB Structure. The COB and COSB each have the format shown in Fig. 10, namely a 4 byte Block Number field 28 (this is the block address of the COB or COSB) and a 32 byte Obsolete or Deleted Sector Mask 30 which is a BitMap containing ones in positions corresponding to obsolete or deleted sectors inside this block. For a block comprising 256 pages this mask 30 takes 32 bytes.

Next Erased Block list (NEB)

[0097] The Next Erased Block list is created in the microprocessor SRAM 13 in order to provide fast identification of the next available erased block when incrementing the WP, SWP and RP between blocks. Available erased blocks are used in ascending order of their physical addresses. The NEB contains M erased block addresses, (e.g. M=8). The NEB list is a list of next available erased blocks, starting with the erased block with block address closest to and higher than the address of the last erased block to be allocated for use. Thus, although the number of entries in the NEB is limited (e.g. to 8), the NEB itself may contain information about more than the next eight erased blocks.

[0098] The Next Erased Block list is derived from the MAP entry (stored in CB) appropriate to the region of Flash memory address space being accessed by the write pointers. This will remain in SRAM as the active NEB until all erased blocks defined by it are used, at which point it must be recreated from the appropriate MAP entries. The CB contains sufficient MAP entries to define the erase state of every block in Flash memory. Both the NEB and the corresponding MAP entry are updated by addition and removal of blocks during operation of the memory system. An entry is removed from the NEB when an erased block is allocated for sector or control data storage. An entry is added to the NEB (unless the NEB is already full) when an erased block is created at a block address which lies within the range spanned by the other blocks in the NEB.

[0099] A single NEB entry defines a group erased blocks at contiguous addresses, and defines the block start address (the first block address in the contiguous group of blocks) and the group length (number of contiguous blocks in the group).

TBL Pointers (TBLP)

[0100] The TBL Pointers identify positions of TBL entries in the CB. They are also stored in the microprocessor SRAM and are used to provide fast sector address translation. The TBLP is created during initial scanning of the Control Block, and then is updated every time a new TBL entry is created in the CB.

ASB List (ASBL)

[0101] As aforementioned, the ASBL is created in SRAM 13 and supports fast address translation. It identifies the physical addresses of all ASB blocks and the SAT blocks with which they are associated. It is created during initialisation and has to be updated each time a SAT page write operation is performed. The ASBL entries are listed in order of previous accesses, that is, when an entry is accessed it is promoted to the top of the list. When a SAT block which does not currently have an associated ASB is accessed, an ASB is allocated and an entry for it is added at the top of the ASBL. The bottom entry of the ASBL, representing the least recently accessed ASB, is eliminated.

SAT Cache

[0102] A cache is maintained in the SRAM 13 for 32 contiguous SAT entries incorporating the entry most recently accessed from a SAT in Flash memory.

Capacity Map

[0103] Total Flash memory capacity in the memory system 10 is allocated to data and control structures as follows :

1. Logical Sectors

Capacity is allocated to storage of one valid data sector for each logical address within the declared logical capacity of the card. This declared capacity is the available physical capacity minus items 2 to 8 below, and is defined by the formatter during card manufacture.

2. Boot Block

At least one block is allocated to the boot block.

Preferably, a second block is allocated for storing another copy of the boot block.

3. Control Block

A number of blocks are allocated to storage of Control Block entries (Signature, BBL, TBL and MAP). The fully compacted Control Block occupies less than one block in most cases. Additional blocks are allocated to the Control Block for entries written between compaction/rewrite operations.

4. Sector Address Table

This is the capacity allocated to the storage of the blocks of the SAT. It is proportional to the logical capacity of the card.

5. Additional SAT Blocks

A fixed number of blocks are allocated for ASBs to be associated with defined SAT blocks.

6. Obsolete Sectors

One block is allocated for the COB. Another block is allocated for the COSB and one further block is allocated for the POB (for embodiments which allow the existence of a COSB and POB). The maximum number of permitted obsolete data sectors is therefore set by the number of pages in a block.

7. Erased Buffer

This is a buffer of erased blocks which must be allocated for correct operation of the system. At least one erased block must be allocated for data sector relocation and one for control structure relocation which may take place concurrently.

8. Spare Blocks

Spare blocks may be allocated for use to maintain declared logical capacity in the event of a failure during operating life. The number of spare blocks is determined by the formatter during card manufacture.

[0104] A Capacity Allocation Table illustrating, for example purposes, capacity allocated to the above items 1-8, for an 8MB card, 64MB Card and 512MB Card (FLASH Cards) is shown in Fig. 16 (this shows only one block allocated for obsolete sectors, namely for the COB, but it will be appreciated that more blocks will be allocated for obsolete sectors where there is provision for a COSB and/or POB in the memory system).

SAT WRITE OPERATIONS

[0105] The SAT is rewritten when the WSL, WBL, SSL, SBL, RSL or RBL, is full and, in the majority of cases, it is done with a granularity of one page, that is, only SAT pages containing modified sector addresses are rewritten. However, in a.small minority of cases, full SAT block rewriting is required (designated "SAT block write"). When a SAT rewrite is required, an Additional SAT Block (ASB) is created, into which only the required pages are written (designated "SAT page write"). There can only be one ASB dedicated to a specific SAT block, and totally, there may exist a limited number N of ASBs. The value of N will be chosen as an appropriate compromise between write performance and capacity overhead (for example, N=8).

[0106] Each ASB exists until a "SAT block write" is performed on its associated SAT block. A SAT block write is performed when a SAT write page is required on a SAT block whose ASB is full or an ASB is to be deallocated. When a SAT page write is required and the corresponding SAT block does not have an associated ASB and all N ASBs are already allocated, one of the existing ASBs has to be deallocated before allocation of a new ASB. The ASB to be deallocated is selected as the one which has not been written for the longest time. Note that deallocation of an ASB is rather time consuming procedure as it requires writing of a SAT block and also erasure of both the obsolete SAT block and ASB.

Interleaved Chip Access

[0107] The above described operations and data structures inherently allow interleaved write operations to be performed on several Flash chips 5 and this can significantly increase performance. The controller chip 8 may thus control a plurality of FLASH chips 5, for example an array of four FLASH chips, incorporated in the memory system. The memory space of the four FLASH chips is organised by the controller as a set of virtual blocks, each of which consists of four blocks 4, one block from each of the four chips 5 (which chips are permanently linked together). The four blocks in each virtual block are the blocks with the same block address within a chip. This organisation allows linked blocks forming a virtual block to be treated as one large block and, therefore, to use all the same described algorithms and data structures as previously described for a single FLASH memory system 10, using virtual blocks instead

of the individual erasable blocks of sectors. Page write operations are performed concurrently across all interleaved chips. The Write Pointer (WP) and the RP and SWP each move sequentially through page addresses, and the ordering of address bits supplied to a hardware chip enable decoder provided in the controller 8 ensures that pages of linked blocks are sequentially accessed in the order illustrated in Fig. 11 i.e. each pointer WP, SWP, RP moves from one chip to another when moving from one PA to another PA. This is achieved by the use of Virtual Addresses. A unique Virtual Address (VA) is allocated to each physical sector (e.g. in the above-described NAND based memory system a VA is allocated for each page) in all of the chips. The Virtual Addresses are allocated such that incrementing the VA by one moves the write pointer from one chip to the next chip, the Virtual addresses incrementing from chip to chip, though linked blocks of each virtual block, in a repeating pattern as shown in Fig. 11.

[0108] The controller in effect treats the array of memory chips as a single column of virtual blocks. The Virtual Address of a sector takes the format shown in Fig. 12. This consists of a Virtual Block portion comprising a ChipHigh portion and the 13 bit Block address of the sector, and a Virtual Page portion Comprising the 6 bit page address and a ChipLow portion. The ChipHigh portion is C_{high} bits of the 5 bit chip number (of the physical address - see Fig. 5) and the ChipLow portion is C_{low} bits of the 5 bit chip number, where:

C_{high} = column number of chip in array of chips; and
C_{low} = row number of chip in array of chips.

To obtain the Physical Address (PA) from the Virtual Address (VA), the controller simply re-organises the VA so as to move the ChipLow portion back between the ChipHigh and Block address portion, as shown in Fig. 13. Thus, it will be appreciated that in a single chip memory system for any sector the VA is equal to the PA.

[0109] For simplicity, only a number of chips which is a binary multiple may be interleaved, for example, 2 or 4. Erase operations on virtual blocks are performed as concurrent erasures of all linked blocks of interleaved chips. If a block in a chip is a bad block the controller treats all the blocks having the equivalent block address in the other chips as bad blocks.

[0110] Interleaving is enabled or disabled according to the status of a control byte in the Boot Block which is set in enabled or disabled status by the manufacturer of the memory system when the FLASH memory is formatted.

[0111] Where block addresses or PAs were previously used in the above-described single chip NAND type FLASH embodiments, we now use Virtual Block addresses and VAs, respectively. On receipt of a host data sector write command the controller translates an incoming LA to a PA by allocating the PA to which the relevant write pointer is pointing. The controller controls the write pointers to each move through the PAs so as to, in effect, move sequentially through the Virtual Addresses (VAs) of the sectors of those of the virtual blocks which are erased (which erased virtual blocks are identified in the NEB).

[0112] Fig. 14 illustrates the timing of the various operations involved in a multiple sector write to interleaved chips. Fig. 14 will now be described with reference to Fig. 15 and Fig.19. Fig 15 is a block diagram illustrating in detail the controller chip 8 of the memory system. (The controller chip 8 of Fig. 2 may be of the form shown in Fig. 15 and like parts in both Fig. 2 and Fig. 15 are numbered with the same reference numerals). Fig. 19 is a schematic diagram of a memory system 10 comprising the controller chip 81 and four FLASH chips 1^1, 2^1, 3^1, 4^1 each with its own read/write buffer 71^1,72^1,73^1,74^1.

[0113] Fig. 15 shows the controller chip 8^1, with : an input/output port O/P e.g. a PC Card ATA Port, Compact Flash or IDE Port, for connection to a host computer; a Host Interface & Registers 80 connected to O/P and a Dual-Port SRAM Sector Buffer 9 connected thereto; A Datapath controller 82, the ECC Generator & Checker 12, and a Flash Memory Interface (FMI) 84, all connected to the Sector Buffer 9, the FMI also being connected to a Flash Memory Port 86. The controller 8 also includes microprocessor 11 (in the form of a RISC Processor); processor SRAM 13,; a processor Mask ROM 88; and a port for an external Program RAM or ROM 92. An optional Debug Port 94 may also be provided for the RISC processor 11. Data and 5 commands are communicated between the various components of the controller 8^1 (except for the Sector Buffer 9) via a microprocessor Bus 91.

[0114] As shown in Fig. 14, when a multiple sector write command (in this case a 4-sector write comprising Host Data sectors 1, 2, 3 & 4) is received at the ATA Port O/P, Sector 1 is written into a Buffer 1 of the Dual-Port Sector Buffer 9. This leaves a Buffer 2 of the Sector Buffer 9 available for controller data management operations. When Sector 2 is received it is written directly into Buffer 2 and at the same time Sector 1 is moved from Buffer 1 to the Flash Memory Port 86 from where it is written into the read/write buffer 71^1 of one of the four FLASH chips (chip 1^1). Sector 2 is then. sent from buffer 2 to the Flash port 86 and on to the read/write buffer 72^1 of one of the other four FLASH chips (chip 2^1). While this is happening Sector 3 is received directly into Buffer 1 of the Sector Buffer 9. Sector 3 is written to the buffer 73^1 of Chip 3 and Sector 4 is received into Buffer 2 of the Sector Buffer 9, and is then written to the buffer 74^1 of chip 4. Sectors 1,2,3 and 4 are then written into the relevant allocated physical sectors in the memory arrays 61^1,62^1,63^1,64^1 of chips 1^1, 2^1, 3^1 & 4^1 respectively. Although Fig. 14 shows each such sector Write operation starting shortly after the previous one, in practice, to all intents and purposes, it will be appreciated that the four Sectors 1,2,3,4 are written substantially concurrently to the Flash Chips 1^1 to 4^1. Moreover it will be appreciated

that the physical addresses of the sectors to which the Host Data Sectors 1 to 4 are written are determined by the algorithms previously described which determine the position of the relevant write pointer (i.e. sequential use of Virtual Addresses).

[0115] It will be appreciated that where a multiple-sector write command of more than four sector writes is sent from the host processor to the controller, the controller partitions the multiple-sector write into groups of (for the present embodiment using four memory chips) four sectors, each such group to be written to FLASH memory in an interleaved write sequence as described above with reference to Fig.14.

Address Translation

[0116] The process of address translation for an LA to a VA will now be described in further detail, with reference to read and write operations.

[0117] Address translation is an operation performed on the logical address of a sector which returns one of the following three values.

- Valid sector physical address
- Information that logical sector has been deleted (or has never been written)
- Information that an error condition has occurred

[0118] Fig. 17 is a flow chart illustrating the address translation process. This process is carried out for every read operation. When a logical sector address (LA) to be translated is received by the controller 8^1 from the host processor an algorithm is implemented (box 40) to identify the possibility that a sector whose logical address is to be translated has previously been written or deleted since its SAT entry was last written. A conclusion that a sector has not been previously written or deleted must be 100% certain; a conclusion that a sector may have been previously written or deleted should have a high probability of being correct. This is carried out by the processor by keeping a limited number of pairs of values representing the start and end addresses of contiguous sequences of sectors, and identifying whether an address to be translated lies within any of these ranges.

These ranges may eventually become non-contiguous, leading to some uncertainty in a conclusion that a sector may have been previously written or deleted. If the LA lies within any of the ranges then we answer "IS Repeat Possible?" (box 42 of Fig.17) with YES. If the LA does not lie within the ranges we answer NO and go to the SAT or SAT Cache to find VA (box 44). From here we determine whether the physical sector is Bad (46) or Deleted (50). If the LA corresponds to an unwritten sector this results in VA=Deleted (58) at box 50. If we answer Yes at box 42, then we search 52 the WSL or SSL (depending on whether the LA corresponds to file or system data). If at box 54 the LA is found (YES) the VA is calculated 56 and the logical address stored in the header 1b is of the physical sector is read by the controller microprocessor (at 58). If LA=LA1 (box 60) then the calculated VA is correct. If the LA is not found at 54 then we search 62 for it in the RSL and if it is not found in the RSL we go to the SAT or SAT Cache and get the VA from there 44. If the VA found in the SAT or Sat Cache is not Bad or Deleted 46, 50, then we get LA1 from the VA 58 and check if LA=LA1, as before.

[0119] The process steps carried out at box 56 (Calculate VA) of Fig.17 are illustrated in detail in the flow diagram of Fig. 20. Fig.20 illustrates the steps carried out in order to obtain the VA for an LA found in the WSL or RSL, for a memory system using only two write pointers , WP and RP. It will be appreciated that this flow diagram would be extended to also allow the VA for an LA found in the SSL, where the memory system also incorporates a System Write Pointer (SWP). The process starts at box 100 where we set NumFromEnd (NFE), where NumFromEnd = number of sectors written beginning from the end of the WSL (or RSL) up to the given sector (found in the WSL or RSL. If the LA was found in the WSL we set P=WP and if LA was found in the RSL we set P=RP; then we set PG= P.Page where P.Page is the page of the write pointer indicated by the value of P (see box 102). If 104 PG > NFE (i.e. LA is in the last written block) then VA= P-NFE-1, namely NFE-1 sectors away from the position of the relevant write pointer. If PG < NFE then we determine 106 if P==0, namely if a block corresponding the last WBL/RBL entry is fully written. If it is (i.e. P==0) we set NotLast=0, and if it is not we set NotLast=1. We then calculate 108 number of blocks, Nblock, between the last one and the block where the given sector lies, using the following algorithm:

Nsect is a number of sectors between last written block page 0 and the given sector;
Nsect= NumFromEnd - PG;
Nblock= NSect/BlockSize + NotLast

[0120] We then calculate 110 page number in a block, PageNum, where PageNum= BlockSize - NSect%BlockSize. If LA is in the WSL we then 112 get Block Address (BLAddr) from the WBL, or if LA is in the RSL we get 114 Block Address (BLAddr) from the RBL, where Block Address is a Virtual address of a block containing the given sector, using the following:

If LA is I WSL, then BlAddr= RBL[LBL - NBlock], where LBL is an index of the last entry in the WBL;
If LA is in RSL, then BlAddr= RBL[LRBL + NBlock], where LRBL is an index of the last entry in the RBL.

[0121] Then we calculate 116 the VA using: VA= Page0 + PageNum, where Page0= Virtual address of page 0 in the block containing the given sector.

[0122] Fig.21 is a flow diagram of the process used to

12/21/2022 19:59:42 Page 16 of 57 SAMSUNG EX1002, Page 230 of 1031

Get VA from SAT or SAT Cache (box 44 in Fig.17). Fig. 21 is generally self-explanatory but further comments regarding specifically labeled boxes is given as follows:

[0123] Box 120 (Is LA in SAT Cache): LA is in the SAT Cache if LA >= FirstCacheEntry. LA <= LA< First-CacheEntry.LA + CacheSize, where FirstCacheEntry. LA = the LA corresponding to the first SAT Entry in the Cache, and (global) CacheSize = number of entries in the SAT Cache;

[0124] Box 122 (Calculate Block and Page in SAT): We calculate SBNum which is a SAT block number for the given LA, and Spage which is a SAT page number for the given LA;

[0125] Box 124 (Calculate number of TBL entry): TBL-Num is a number of the required TBL Entry, where TBL-Num = SBNum/8; and

[0126] Box 126 (Store part of SAT page in cache): If it is possible, 32 entries starting with the last entry accessed are cached. If there are not enough entries then a group of 32 entries ending with the last entry in a page and including the last sector accesses is cached.

[0127] Fig.18 is a flow diagram illustrating the process steps for Box 58 (Get LA1 from VA) of Fig.17. It should be noted that the Header Parameter (HP) stored in the Page Header will be the value of the logical address (LA) incremented by one. This is because deleted sectors are marked by setting all bits in their headers to zero. This LA=0 cannot be stored in a header. We therefore set LA1= HP - 1.

Read Operations

[0128] Fig.22 is a flow diagram illustrating the sequence of steps carried out to read a host data sector from a physical sector. The controller starts by translating the LA (received from the host) to a VA (box 130). This is done by carrying out the process illustrated in the Fig.17 flow chart, already described. Once the LA has been translated to a VA, the content of the physical sector with address VA is read into the buffer 9 of the controller. The controller then checks (box 132) if the sector (i.e. the content of the sector) is deleted or bad. If it is deleted or bad, the controller sets all bytes of the host processor's data buffer to OxFF (box 134). If the sector is not a deleted sector (box 136), the controller returns error status to the host (box 138). If the sector is a deleted sector, the controller returns valid status to the host (box 137). If at box 132 the controller determines that the sector is not deleted or bad, the controller goes straight to box 137 i.e. returns valid status to the host.

Write Operations

[0129] Fig.23 is a flow diagram illustrating the sequence of steps carried out to write a host data sector to a physical sector.. Fig.23 deals only with write operations for host file data, and thus written by the Write Pointer (WP), but it will be appreciated that the operations of Fig.23 would be appropriately extended to deal with separate system data writes where the memory system uses a separate write pointer for system data (i. e. the SWP). The controller starts by translating the LA (received from the host) to a VA (box 150). This is done by carrying out the process illustrated in the Fig.17 flow chart, already described. If 152 the sector is Bad, the controller returns error status 154 to the host. If the sector is not Bad, then check if the sector is deleted 156 and if it is deleted then check if WP is valid or invalid 158. WP is invalid (WP==0) when a full block has just been written and WP has to be moved to an erased block. If WP is not valid, we set the WP to a new (valid) physical sector address 160. When WP is valid, we add the LA to the WSL 162 and perform any WSL or RSL release, and/or CB and CBPT compaction, which is necessary. We then update 164 the ranges from the Evaluate Repeat Possibility algorithm (box 40 of Fig.17) and write the sector 166 from the controller buffer to the address of the WP, and return a valid status value 168 to the host. If, at box 156, we find the sector is not deleted we then check 157 if the VA is in the COB. (The VA is in the COB if VA coincides with VB, where VB is a Virtual Block Number field (this is the Virtual Block address - see Fig.12) in the COB structure stored in SRAM 13. If the VA is in the COB, we record VA as obsolete 159 in the COB structure stored in the controller SRAM 13 (this is done by setting a corresponding bit to 1 in the BitMask field of the COB Structure in SRAM 13.) and then go on to box 158 (is WP valid). If the VA is not in the COB we change the COB 161 and then move on to box 159 (Mark VA as Obsolete).

[0130] Fig.24 is a flow diagram of the steps implemented at box 161 of Fig.23 (Change the COB). Fig.24 is generally self-explanatory but should be read in conjunction with the following notes:

Box 200 (VA.B1, VB): VA.B1 is a Virtual Block field of VA, and VB is as above-described;

Box 202 (Is COB invalid): COB is invalid if VB=0. If VB is equal to zero this indicates that there are no obsolete data at this moment;

Boxes 203,204 (Calculate MaxRel): MaxRel is a maximum number of sectors to be relocated from COB. MaxRel=P.Page-1, where P.Page is a Page field (address) of the WP or RP;

Boxes 205,206 (Add Dummy Entries to WSL): If a block to be relocated is not fully written yet, corresponding "dummy" sector LAs must be added to the last WSL (RSL) Entry;

Box 207 (Relocate Sectors): See Fig.25;

Box 208 (Write WMAP): Write WMAP to the CB where EB+VB and corresponding bit in the BitMap is set to 1. Perform CB rewrite if necessary;

Box 209 (Update Lists): Find WRBArray entry equal to VB and mark it and any other entries for the same VB in WRBArray as invalid. The WRBArray is in fact the WBL and WSL lists which are actually stored in

the same area of memory with the WBL entries at the start counting up and the RBL entries at the end counting down. The WRBArray is full when the two lists meet in the middle.

Box 210 (Setup COB): Update COB Structure in SRAM, VB field is set to VA.B1, Obs and Del Mask bit corresponding to VA.Page is set to 1, all other bits are set to 0.

[0131] Fig.25 illustrates the steps implemented at box 207 (Relocate Sectors) of Fig.24. This should be read in conjunction with the following notes:

Boxes 220, 222 and 230: Perform a loop going through Obs and Del Mask field of the COB Structure in SRAM;

Box 223 (Is Sector Valid?): Sector is valid if ODMask [i]=0; zero value in COB Obs and Del Mask indicates that this page contains a valid sector;

Boxes 224 and 225: RP=0 if a block pointed to by the RP is already fully written;

Box 226 (Store LA from Page Header): LA got from Page Header is temporarily stored to be used in Add Entry to RSL;

Box 225 (RP=0): Add Entry to the RSL. Perform WSL/RSL release, if necessary.

[0132] Fig.26 illustrates the steps implemented in order to set the Write Pointer (WP), at box 160 in Fig.23. A similar process is used to set the RP at box 227 in Fig. 25. Fig.26 should be read in conjunction with the following comments:

Box 240 (Is WRArray not full): WRArray is not full if Last<LastRE-1, where Last (global) is an index of the last WSL entry, and LastRE is an index of the last RSL Entry;

Box 242 (Is WRBArray not full): WRBArray is not full if LBL<LRBL-1, where LBL and LRBL are as previously defined with regard to Fig.20;

Box 244 (Lists Release): Perform SAT Page Write operation, SAT Block Write and CB rewrite if necessary;

Box 246 (Fill NEB): Select next N (N=NEBSize) erased blocks from MAPs stored in the CB;

Box 248 (Write WMAP): Write WMAP with Link field set to ErBlock and corresponding bit in BitMap field set to 0.

Perform CB rewrite if necessary.

[0133] Fig.27 is a flow diagram of the process steps carried out in a delete sector operation. This is in the initial steps similar to a write operation (see Fig.23): the LA is translated 250 to a VA. If 252 the sector is deleted we return error status to the host 254. If the sector is not deleted we check 256 if the VA is in the COB and if it is not 261 in the COB we Change the COB 261 (same as Fig.24). If VA is in the COB we Mark VA in the COB as deleted 269, then we fill one of the buffers of the Dual port SRAM 9 (in the controller) with zeros 271, and then write this "all zeros" page from the buffer to the sector to be deleted (thereby deleting the sector). We then return valid status 275 (confirming sector has been deleted) to the host.

Initialisation

[0134] Initialisation procedures and power-loss recovery procedures will now be described. For simplicity, these are described with reference to a single write pointer system (i.e. only WP) but it will be appreciated that the procedures will be readily extended as appropriate for the multiple write pointer system (WP, SWP, RP).

[0135] All data and control structures are specially constructed to avoid generalised scanning during initialisation. Almost all control structures (except WSL and WBL) normally are derived from corresponding information stored in the CB. During initialisation of the card, it is necessary to perform following operations.

1. To read the last Control Block Pointers Table entry from the Boot Block and so identify the CB block (s) locations.
2. To reconstruct the TBLP by scanning the CB.
3. To scan header/ECC fields of pages sequentially following the write pointer position defined in the last TBL entry in the Control Block to identify sectors written since the last SAT rewrite and to construct the WSL and WBL.
4. To construct the NEB from corresponding MAP entries in the CB.
5. To construct the COB and ASBL.
6. To check if a block referenced in the ErB field of the last MAP is really erased. If not, to complete erase operation.

Construction of WSL and WBL

[0136] During initialisation of the card, the last value of the Write Pointer (WP) to be stored is read from the latest TBL entry in the CB and a scan of page headers at successive pages from that location is performed to rebuild the WSL and WBL in processor SRAM. When an erased location is encountered, the end of the sequence of sectors written since the last SAT rewrite has been reached.

[0137] This sector scan must take account of the fact that the Write Pointer (WP) may jump from the end of a block to the beginning of a non-adjacent block. All block transitions made by the (WP) are recorded in the Link fields of MAP entries in the CB.

Construction of COB and ASBL

[0138] These structures can be reconstructed by cop-

ying corresponding entries from the CB. In addition, to construct the COB structure (to identify deleted sectors in it) it is necessary to scan a current block containing these sectors, whose address is defined in the Obs field of the last MAP entry in the CB. To identify obsolete sectors in this block, it is also necessary to scan WSL and WBL. In order to record ASBL pages we have to identify ASB addresses from the TBL and then to scan their header/ECC fields.

Power-Loss Recovery

[0139] It is a requirement for the memory system that it should be able to operate normally, and that no stored data should be lost, when power is restored, whatever the circumstances under which power has been removed. However, it is not necessary to restore the full normal state of the memory system immediately after power-on, only to allow it to operate normally. A normal state can be restored later as. an exception, whenever any abnormal state is detected.

[0140] The normal state of the Memory System may be degraded if the supply voltage is removed whilst any of the following operations is being performed.

1. Writing of a data sector from a host
2. Writing of a data sector which is being relocated
3. Writing of an entry to a control data block (CB or BB)
4. Writing of a page to a control data block (SAT or CB)
5. Erasure of any block with obsolete sector or control data

Power loss during writing of a data sector from a host

[0141] In this case, the data being written may be lost, but the host had not been informed that the write command had completed and may write the sector again. An incompletely written sector may exist in Flash memory as a result of the partial write operation. This is detected during initialisation when the value of the Write Pointer is established by reading the page headers in the block defined by the last Link parameter in the CB. The last detected sector should read fully to check its ECC, and the next page should be read to check that it is completely erased. If a partially written sector is detected, all other sectors should be relocated to a new Write Pointer position at the beginning of the next erased block, then the block should be erased.

Power loss during writing of a data sector which is being relocated

[0142] This is detected during the process of establishing the Write Pointer during initialisation, as above. The same action of relocating sectors and erasing the block should be taken. In addition, an incomplete relocation operation should be detected by comparing logical sector addresses immediately preceding the Write Pointer with those of obsolete sectors in the block defined by the Obs parameter in the CB. Any pending sector relocations should be completed during initialisation.

Power loss during writing of an entry to a control data block (CB or BB)

[0143] This condition may be detected during normal initialisation when entries in the CB and BB are read and their ECCs are checked. The entry should be ignored if its ECC shows an error. The earlier operation which initiated the CB or BB entry write operation had not completed correctly and the entry will later be written correctly when this operation is repeated during normal operation.

Power loss during writing of a page to a control data block (ASB)

[0144] This condition may be detected during normal initialisation when pages in the ASB are read and their ECCs are checked. The page should be ignored if its ECC shows an error. The earlier operation which initiated the ASB page write operation has not completed correctly and the page will later be written correctly when this operation is repeated during normal operation.

Power loss during writing of a full control data block

(SAT or CB)

[0145] This will result in an incomplete control data block existing in Flash memory, with no references to it by other data structures. This condition need not be detected during initialisation, and the block may be allowed to exist as a "lost block". The earlier operation which initiated the block write operation had not competed correctly and the block will later be written correctly when this operation is repeated during normal operation. At a later stage of normal operation, the lost block will be detected by a discrepancy with its MAP state, or by the discovery of a discrepancy in the number of erased blocks in the system (see Capacity Map in Fig. 16). Exception routines may then identify and erase the block, by full FLASH memory scanning if necessary.

Power loss during erasure of a block with obsolete sector or control data

[0146] This will result in an incompletely erased block existing in Flash memory. This condition is detected during initialisation when the state of the block referenced by the ErB field in the last MAP entry in the CB is checked. Re-erasure of this block can be performed, if necessary.

12/21/2022 19:59:42 Page 19 of 57 SAMSUNG EX1002, Page 233 of 1031

Further Alternative Embodiments

[0147] Various modifications to the above-described embodiments are possible without departing from the scope of the invention. For example, one alternative way of handling erasure operations is to always allow two COBs (and two COSBs) to exist: the advantage of this would be to make the best use of memory capacity. In the above-described embodiment, we only allow one COB, but also allow a POB to exist temporarily when there is a write pointer in a block which we wish to make the COB. This means that there must at all times be enough erased memory capacity to allocate for a POB, should it be necessary to have a POB. It therefore is attractive to make the best use of this memory capacity and one way of ensuring this is to always allow two COBs to exist, therefore eliminating the need for a POB (the second COB can act as a POB, when required). In such a two COB system, when it becomes necessary to create a new COB we erase the older one of the two COBs (unless it has a write pointer in it in which case we erase the younger one).

[0148] With reference to Fig.3, and the description of the arrangement of data in each of the FLASH pages in the memory system, we also propose some alternative ways of storing the data within a page. Fig.28 shows the physical partitioning of a typical 528 Byte NAND or AND type FLASH memory page 1. The page comprises a 512 Byte "data area" 300 and a 16-Byte "Spare area" 302. In the embodiment described above with reference to Fig.3, the controller 8 stores 512Bytes of information 1a (e.g. one host data sector) in the Data Area 300, and stores the Header 1b and ECC 1c (together referred to hereinafter as Overhead Data (OD)) in the Spare Area 302. However, other arrangements of data within the page 1 are possible. For example, as shown in Fig.29, the Header 1b and ECC 1c could equally be stored in a first portion 303 of the Data Area 300, and the Information 1a stored in the portion 304 consisting of the Spare Area 302 and the remainder of the Data Area 300.

[0149] Another possibility, shown in Fig.30, is to write the Header 1b and ECC 1c at a position offset from the start of the FLASH page, and to write the Host Data Sector (which may be referred to as the "user data") in the remaining space either side of the Header and ECC. By how much (Offset S) the OD is offset may, for example, be determined by a function of either: (a) the physical address (PA) of the page 1; or (b) one or more bits within the first byte of user data (i.e. the host data sector) written to the page 1. Fig.31(a) illustrates the arrangement of data in the controller buffer 320 before the start of a sector write operation, the data being arranged as a first portion 322 of user data and a second portion 324 of Header data. Fig.31(b) shows the arrangement of the data in a FLASH memory page, following completion of the write operation in which the offset S is determined by one or more bits within the first byte of user data (option (b) above). The data is stored as a first portion 326 of the user data, followed by a second portion 328 of the user data, followed by the Header 1b and ECC 1c, followed by the third and final portion 330 of the user data. The length of portion 326 + portion 328 is dependent on. data within portion 326. The length of portion 326 is defined to be less than or equal to the minimum offset, and the length of 328 is calculated on the basis of data within portion 326 to provide the correct Offset S. The first and second portions 326,328 of user data are separately identified so that the first portion 326 may be read from the FLASH memory by the controller in one operation, and evaluated by the controller in order to determine the length of the second portion 328 which should be read by the controller in a subsequent operation Fig.32 is a table detailing the sequence of controller commands used to transfer the data from the controller buffer to the FLASH memory during the write operation.

[0150] One advantage of choosing the offset S to be a function of one or more bits of the user data is that the overhead data is therefore not inserted at the same position in the 528Byte data segment in every sector. This protects the valuable overhead data from simultaneous loss on a large number of sectors in the event of a systematic failure in FLASH memory, such as a memory array column failure.

[0151] Fig.33 shows the resulting arrangement of data in the controller buffer after completion of a read operation on the FLASH memory page of Fig.31(b). From Fig.33 it will be seen that the data in the buffer is arranged back to a first portion 322 of user data and a second portion 324 of Header data, now followed by a third and final portion 325 of ECC. Fig.34 is a table detailing the sequence of controller commands used to transfer the data from the FLASH memory to the controller buffer during the read operation.

[0152] Additionally, with reference to interleaved write operations to multiple FLASH chips, as described already with reference to multiple FLASH chip memory systems, we also propose that this technique for writing substantially concurrently to a plurality of chips may also be used for writing data to a single memory chip in which the physical page size is a multiple of the size of a sector write by the controller e.g. each page of the memory is four times the size of a segment of (user + overhead) data written by the controller, where the controller writes data in uniformly sized segments.

[0153] It will further be appreciated that the invention is applicable not only to NAND-type memories but also to other types of memory, such as AND and NOR type memories. In the case of AND type FLASH memory, each page 1 of a block has the same format as the NAND page format of Fig.28 and we can use any of the possible arrangements of data within the pages as afore-described. We design the controller to still erase memory in blocks of sectors, although in blocks containing bad sectors the individual good sectors in the block to be erased will be erased individually. Thus, the controller does not treat any blocks containing bad sectors

as bad blocks, but instead treats them as good (erasable) blocks and makes use of the good sectors within those blocks. In AND type embodiments though we ensure that the controller only uses blocks containing all good sectors for SAT blocks or ASBs.

[0154] Where AND type FLASH memory is being used and the memory system is a multiple FLASH chip system utilizing interleaved chip write operations as described above, where any block of sectors (pages) in one of the virtual blocks contains a bad sector, the controller causes the write pointers to skip this sector and go to the next good sector in the block e.g. where c=chip and s=sector, if a burst of four sector writes is c3s5, c4s5, c1s6, c2s6 then if c1s6 is a bad sector the sequence becomes c3s5, c4s5, c2s6, c3s6. This is in contrast to embodiments based on NAND type memory, where if one block in a virtual block contains one or more bad sectors the controller treats that block as a bad block and treats the whole virtual block as a bad virtual block.

[0155] Where we use NOR type FLASH memory, our preferred embodiment is one in which we design the controller of the memory system to still read and write data structures to and from the FLASH memory in uniformly sized sectors, each sector being 528 Bytes wide. Fig.35 illustrates schematically three such sectors 1,2,3 in a block 4' of NOR memory. Due to the fact that one row of memory in a NOR block is only 512 Bytes wide it will be appreciated that each of our sectors in NOR therefore fills one row and wraps round to fill a part of the next row. Nevertheless, it would be possible to define our sectors in NOR memory in a different manner, and we may choose to use sectors of smaller or larger size than 528 Bytes, and a block could even contain sectors of more than one size. The controller may arrange the data within each sector in any of the various different ways already described with reference to NAND and AND type memory pages, each sector including user and overhead data.

[0156] It will be appreciated from the foregoing that the physical sectors of the memory system, whether the memory system is based on NAND, AND or NOR type memory arrays, need not have any particular relationship to the physical architecture of the memory array itself, for example a sector need not correspond to a row (page) of the array, and no physical feature need be present to identify the boundary between one sector and a neighbouring sector. A sector can be interpreted as a group of memory cells which are always treated by the controller as a unit. Different sectors need not be of the same size. Moreover, the physical structure of a sector has no dependence on data which might be stored in the sector. Also, embodiments are possible in which defective regions within a row (page) of memory cells are tolerated and are simply skipped over by the controller when writing to physical sectors.

[0157] With reference to the SAT, while as above-described the SAT is preferably stored in one or more blocks of the memory array 6, it would alternatively be possible to provide in the memory system 10 a separate non-volatile memory which is accessible to the controller, in which separate memory the controller stores the SAT.

[0158] Finally, in a modified version of the above-described embodiment, instead of always using available erased blocks in ascending order of their physical addresses as above-described, the controller uses the erased blocks in another order. In this modified embodiment, the NEB list contains a chosen subset of all the currently available erased blocks, the first block address in the NEB list is the next erased block to be used, and this first block address is removed from the NEB list when it has been allocated for data storage use. Any new erased block which is created (e.g. due to creation of obsolete data, following a delete command from the host) is added to the bottom of the NEB list. This continues for a period determined by the controller (which could be a predetermined number of sector write commands from the host, for example), at the end of which period the controller re-compiles the NEB list by replacing the entries in the NEB with a new subset of the currently available erased blocks. Conveniently, subsets of the whole set of all erased blocks may be used sequentially in order of ascending physical block addresses. This modified embodiment may have some advantage in reducing memory space requirements in connection with monitoring and storing the erased state of all blocks.

Claims

1. A memory system (5,8) for connection to a host processor, the system comprising:

   a solid state memory (5) having non-volatile memory sectors (1,2,3) which are individually addressable and which are arranged in erasable blocks (4) of sectors, each said sector having a physical address defining its physical position in the memory;
   and a controller (8) for writing data structures to and reading data structures from the memory, and for sorting the blocks of sectors into blocks which are treated as erased and blocks which are treated as not erased; wherein the controller includes:

      means (11) for translating logical addresses received from the host processor to physical addresses of said memory sectors in the memory;
      a write pointer for pointing to the physical address of a sector to which data is to be written from the host processor, said write pointer being controlled by the controller to

12/21/2022 19:59:42 Page 21 of 57 SAMSUNG EX1002, Page 235 of 1031

move in a predetermined order through the physical addresses of the memory sectors of any block which is treated as erased and, when the block has been filled, to move to another of the erased blocks;

wherein the controller is configured so that, when a sector write command is received from the host processor, the controller translates a logical address received from the host processor to a physical address to which data is written by allocating for said logical address that physical address to which said write pointer is currently pointing, and wherein the controller is configured to compile a sector allocation table of logical addresses of sectors with respective physical addresses of sectors which have been allocated therefor by the controller, and wherein the controller updates the sector allocation table less frequently than memory sectors are written to with data from the host processor, **characterised in that** there is provided at least one additional block of sectors containing modified versions of individual sectors of a sector allocation table block, and wherein, when all the sectors in a said additional block are written to with modified versions of sector allocation table sector(s), the respective sector allocation table block is rewritten so as to include all the modified versions in the additional block and the additional block is erased.

2. A memory system according to claim 1, wherein said write pointer is controlled by the controller to move in a predetermined order through the blocks which are treated as erased.

3. A memory system according to claim 1 or claim 2, wherein the physical sector addresses in the sector allocation table are ordered by logical sector addresses, whereby the Nth sector allocation table entry contains the physical address of a sector to which data having logical address N has been written.

4. A memory system according to claim 3, wherein the controller is configured so that when a sector read command is received from the host processor the controller looks up a logical sector address received from the host processor in the sector allocation table in order to obtain the physical sector address which the controller previously allocated to said logical sector address.

5. A memory system according to any preceding claim, wherein the sector allocation table is stored in at least one of said blocks of memory sectors in the solid state memory.

6. A memory system according to claim 5, wherein the controller is configured to update the sector allocation table by rewriting the sector allocation table in whole blocks.

7. A memory system according to claim 1, wherein each sector in a said additional block contains the physical address of the sector of the sector allocation table block which it updates, and the modified version of the said sector of the sector allocation table block.

8. A memory system according to any preceding claim, wherein the controller is configured to control the write pointer so as to move sequentially, in ascending numerical order of physical address, through the erased blocks, as each block is filled with data written thereto.

9. A memory system according to claim 8, wherein the control of the write pointer is cyclic in the sense that once the sectors in the highest block, according to physical address order, have been filled with data the write pointer is controlled by the controller to wrap around to the block of sectors having the numerically lowest physical block address out of all the blocks currently being treated by the controller as erased.

10. A memory system according to any of claims 1 to 7, wherein the controller is configured to control the write pointer to move non-sequentially, according to physical address order, through the erased blocks.

11. A memory system according to any of claims 1 to 10, wherein each said memory sector is physically partitioned into a data area (300) and a spare area (302) and the controller is configured so as to write overhead data comprising header data and error correction code data at a position in the sector which is offset from the start of the data area of the sector and to write user data, received from the host processor, in the space remaining in the sector, on either side of the overhead data.

12. A memory system according to claim 11, wherein said overhead data is offset by an amount which is determined by at least one bit of the user data to be written to the sector.

13. A memory system according to any preceding claim, wherein the memory sectors in each said block of sectors are erasable together as a unit.

14. A memory system according to claim 13, wherein the memory sectors in each said block of sectors are also individually erasable.

15. A memory system according to any preceding

12/21/2022 19:59:42 Page 22 of 57
SAMSUNG EX1002, Page 236 of 1031

claim, wherein the controller is configured to control erase operations on the memory so as to only erase whole blocks of memory sectors, and wherein a block of sectors is treated by the controller as an erased block if all the memory sectors therein are erased sectors.

16. A memory system according to claim 15, wherein if a block contains one or more bad sectors, the controller defines the whole block as being bad and treats that block as a not erased block, whereby no data will be written thereto.

17. A memory system according to claim 14, wherein if a block contains one or more bad sectors the controller treats that block as an erased block whereby the controller may still use good sectors in the block to store data, and wherein the memory system includes a table identifying bad sectors and the controller is configured to check whether the next sector address to which the write pointer is to be moved is the address of a bad sector and, if it is the address of a bad sector, to control the write pointer to skip this bad sector and move to the next sector address according to the predetermined order in which the sectors are to be written to.

18. A memory system according to any preceding claim, wherein each block of sectors has a physical block address defining its physical position in the memory and the physical address of each said memory sector includes the physical block address of the block in which it is located, and wherein the controller is configured to compile a list of the physical block addresses of at least some of the blocks of sectors being treated as erased, listed in an order in which the write pointer is to move through the blocks, which list is used bythe controller in order to quickly identify the next block of sectors to be written to, and the memory system further includes temporary memory means in which said list is stored by the controller.

19. A memory system according to any preceding claim, wherein the controller is configured so that, when a sector write command is received by the controller from the host processor which command renders obsolete data previously written to another sector, the controller stores in a temporary memory of the memory system the address of the sector containing the now obsolete data.

20. A memory system according to claim 19, wherein the controller is further configured so that if a sector delete command, generated by a user, is received from the host processor by the controller, the controller marks as obsolete the sector to be deleted and stores the address of the sector in said temporary memory.

21. A memory system according to claim 19 or claim 20, wherein the controller is configured so as to allow only a fixed predetermined number of blocks at any time, herein referred to as the current obsolete blocks, to contain one or more sectors containing obsolete data which was written by the write pointer, and so that when all the sectors in a said current obsolete block contain obsolete data, the said current obsolete block is immediately erased.

22. A memory system according to claim 21, wherein the controller is configured so that where a sector in a block other than a said current obsolete block is to contain obsolete data, the controller: relocates any data in valid (not obsolete) sectors in a said current obsolete block to another block and then erases the said current obsolete block; marks said sector in the said block other than a current obsolete block as obsolete; and designates said other block as a new current obsolete block.

23. A memory system according to claim 21, or claim 22, wherein said fixed predetermined number of current obsolete blocks is one.

24. A memory system according to claim 22, wherein said block to which the controller relocates said valid data is the block in which the write pointer is currently located.

25. A memory system according to claim 22, wherein the memory system includes a further write pointer, herein referred to as the relocation pointer, for pointing to the physical address of the sector to which said valid data is to be relocated, the relocation pointer always being in a different block of sectors to the write pointer.

26. A memory system according to claim 25, wherein the memory system includes a further write pointer, referred to as the system write pointer, which points to the physical address of the sector to which system data is to be written from the host, the system write pointer always being in a different block to the write pointer.

27. A memory system according to claim 26, wherein the controller is configured so as to allow at least two blocks which contain one ore more obsolete sectors to exist at any time, one being said current obsolete block and the other being a current obsolete system block containing one or more obsolete system data sectors and, if any system data sectors need to be relocated in order to allow the current obsolete system block to be erased, the relocated system data is sent to the address to which the sys-

12/21/2022 19:59:42 Page 23 of 57 SAMSUNG EX1002, Page 237 of 1031

tem write pointer is currently pointing.

28. A memory system according to claim 26, wherein the memory system includes another write pointer, herein referred to as the system relocation pointer, for pointing to the physical address of the sector to which valid system data is to be relocated, the system relocation pointer, always being in a different block of sectors to the write pointer and the system write pointer.

29. A memory system according to any of claims 26 to 28, wherein the controller is configured to that if the current obsolete block contains one of said write pointers at the time when the controller needs to erase a said current obsolete block because obsolete data has just been created in another block, the controller proceeds with creating a new current obsolete block but postpones the erasure of the old current obsolete block, herein referred to as the pending obsolete block, until all erased sectors in the pending obsolete block have been filled and said pointer moves on to the next erased block to be used, as defined by the controller, at which time any valid (not obsolete) data in the pending obsolete block is relocated by the controller and the pending obsolete block is erased.

30. A memory system according to claim 27, wherein the controller is configured to store in a temporary memory of the memory system respective lists of logical sector addresses corresponding to sectors in the memory to which relocated data has been written to by the relocation pointer herein referred to as the relocation sector list, the system write pointer herein referred to as the write system sector list, and the system relocation pointer herein referred to as the system relocation sector List since the sector allocation table was last updated, and the controller is configured to store in said temporary memory corresponding lists of the order of blocks which have been used by the relocation pointer, system write pointer and system relocation pointer herein referred to as the relocation block list, the write system block list and the system relocation block list (SRBL).

31. A memory system according to any preceding claim, wherein in addition to writing data structures to the memory from the host processor, the controller also generates and writes to the memory data designated as control information, and the controller is configured so as to write such control information in one or more different control blocks of the blocks of memory sectors to those in which data structures received from the host processor are written.

32. A memory system according to claim 31, wherein the controller is configured to store in at least one said control block a list of the block addresses of all the sector allocation table blocks.

33. A memory system according to claim 31 or claim 32, wherein the controller is configured to store the block addresses of said one or more control blocks in a dedicated boot block of the memory, this boot block being a dedicated block being the first block of sectors in the memory which does not contain any bad sectors.

34. A memory system according to claim 32, as dependent from claim 1 or 7, wherein said list of all the sector allocation table block addresses is in the form of a plurality of list portions and each said portion contains the block addresses of a group of logically contiguous sector allocation table blocks and any additional sector allocation table blocks corresponding thereto.

35. A memory system according to any of claims 1 and 8, wherein the controller is configured to store in a temporary memory of the memory system a write sector list of logical sector addresses for data structures which have been written by the write pointer since the SAT was last updated.

36. A memory system according to claim 35, wherein the controller is configured to also store in said temporary memory the order in which blocks have been used by the write pointer for writing data since the last update of the sector allocation table, this order being stored in the form of a write block list of block addresses of the blocks in which the updated sectors whose addresses are held in the write sector list are located.

37. A memory according to claim 36, wherein the write sector list has a predetermined size and once the Write Sector List is full at least one sector allocation table block or additional sector allocation table block is updated and the write sector list and write block list are emptied.

38. A memory system according to claim 36, wherein the controller stores a starting physical sector address, and the links between blocks containing sectors to which data has been written by the controller since the last update of a sector allocation table or additional sector allocation table block, in a said control block of the solid state memory.

39. A memory system according to any preceding claim, wherein each said sector consists of a single "page" of memory, namely one row of memory cells in a said block of memory sectors.

12/21/2022 19:59:42 Page 24 of 57
SAMSUNG EX1002, Page 238 of 1031

40. A memory system according to any preceding claim, wherein the controller is configured to write data to, and read data from, the memory sectors in uniformly sized data segments.

41. A memory system according to claim 40, wherein all the memory sectors are the same size and each said data segment is equal in size to the size of a said memory sector.

42. A memory system according to any preceding claim, further including a temporary cache memory in which the controller is configured to store a group of contiguous sector allocation table entries including the sector allocation table entry most recently accessed from the sector allocation table by the controller.

43. A memory system according to claim 41, when dependent from claim 7, wherein the controller is configured to create in said temporary cache memory a list of physical addresses of all additional sector allocation table blocks and the sector allocation table blocks with which they are associated which is updated each time a sector allocation table sector write operation is performed.

44. A memory system according to any preceding claim, wherein the solid state memory comprises a single memory array in the form of a single memory chip.

45. A memory system according to any of claims 1 to 43, wherein the solid state memory comprises a memory array formed by a plurality of memory chips.

46. A memory system according to any of claims 1 to 43, wherein the solid state memory comprises a plurality of memory arrays in the form of a plurality of memory chips, and wherein the controller is configured to form the memory sectors in the plurality of memory chips into a multiplicity of virtual blocks, each said virtual block comprising one erasable block of memory sectors from each said memory chip, and to sort said virtual blocks into ones which are treated as erased and ones which are treated as not erased.

47. A memory system according to claim 46, wherein the controller is configured to compile a list of the virtual blocks treated as erased and store this in temporary memory in the memory system, and to control the write pointer to move from one chip to another for each consecutive sector write operation, starting at one sector in one erasable block of the virtual block and moving consecutively to one sector in each of the other erasable blocks in the virtual block until one sector has been written in each erasable block of the virtual block, and then moving back to the chip in which the first sector was written and proceeding in a similar manner to fill another one sector in each erasable block of the virtual block, and so on until the virtual block is full of data, and then to move the write pointer on to the next virtual block in said list of virtual blocks being treated as erased, and fill this next virtual block in a similar manner.

48. A memory system according to claim 47, wherein the controller is configured so that for every n contiguous sector write operations the controller executes for a multiple sector write command received from the host processor, where n is less than or equal to the number of solid state memory chips in the memory system, the controller writes substantially concurrently to one sector in each of n of the chips.

49. A memory system according to claim 47 or claim 48, wherein the controller is configured to carry out erasure of any said virtual block by concurrently erasing all the erasable blocks in the virtual block.

50. A method of controlling reading and writing of data structures to and from a solid state memory (5) having non-volatile memory sectors (1,2,3) which are individually addressable and which are arranged in erasable blocks (4) of sectors, each said sector having a physical address defining its physical position in the memory, the method comprising the steps of:

sorting the blocks of sectors into blocks which are treated as erased and blocks which are treated as not erased;

providing a write pointer for pointing to the physical address of a sector to which data is to be written from a host processor, and controlling said at least one write pointer so as to move in a predetermined order through the physical addresses of the memory sectors of any block which is treated as erased, and when the block has been filled to move to another of the erased blocks and, when a sector write command is received from the host processor, translating a logical address received from the host processor to a physical address to which data is written by allocating for said logical address that physical address to which said write pointer is currently pointing;

storing in non-volatile solid state memory a sector allocation table of logical addresses of sectors with respective physical addresses of sectors which have been allocated therefor by a controller (8);

and updating the sector allocation table less

12/21/2022 19:59:42 Page 25 of 57
SAMSUNG EX1002, Page 239 of 1031

frequently than memory sectors are written to with data from the host processor;

characterised in that there is provided at least one additional block of sectors containing modified versions of individual sectors of a sector allocation table block, and wherein, when all the sectors in a said additional block are written to with modified versions of sector allocation table sector (s), the respective sector allocation table block is rewritten so as to include all the modified versions in the additional block and the additional block is erased.

Patentansprüche

1. Speichersystem (5, 8) zur Verbindung mit einem Host-Prozessor, wobei das System aufweist:

einen Festkörperspeicher (5) mit nicht-flüchtigen Speichersektoren (1, 2, 3), die einzeln adressierbar und in löschbaren Sektorenblökken (4) angeordnet sind, wobei jeder Sektor eine physikalische Adresse aufweist, die seine physikalische Position in dem Speicher definiert; und
eine Steuereinheit (8) zum Schreiben und Lesen von Datenstrukturen in den bzw. aus dem Speicher, und zum Speichern der Sektorenblöcke in Blöcke, die als gelöscht behandelt werden, und Blöcke, die als nicht gelöscht behandelt werden; wobei die Steuereinheit aufweist:

eine Einrichtung (11) zum Übersetzen von von dem Host-Prozessor empfangenen logischen Adressen in physikalische Adressen der Speichersektoren in dem Speicher;
einen Schreibezeiger, der auf die physikalische Adresse eines Sektors zeigt, in den die Daten von dem Host-Prozessor zu schreiben sind, wobei der Schreibezeiger durch die Steuereinheit so gesteuert wird, daß er sich in einer vorbestimmten Reihenfolge durch die physikalischen Adressen der Speichersektoren eines Blocks bewegt, der als gelöscht behandelt wird, und sich, wenn der Block gefüllt ist, zu einem anderen gelöschten Block bewegt;

wobei die Steuereinheit dazu ausgelegt ist, wenn ein Sektor-Schreibebefehl von dem Host-Prozessor empfangen wird, eine von dem Host-Prozessor empfangene logische Adresse in eine physikalische Adresse zu übersetzen, in die Daten geschrieben werden, indem der logischen Adresse die physikalische Adresse zugeordnet wird, auf die der Schreibezeiger gegenwärtig zeigt, und wobei die Steuereinheit dazu ausgelegt ist, eine Sektorzuordnungstabelle von logischen Adressen der Sektoren mit den zugehörigen physikalischen Adressen der Sektoren, die ihnen durch die Steuereinheit zugeordnet wurden, zusammenzustellen, und wobei die Steuereinheit die Sektorzuordnungstabelle weniger häufig aktualisiert als Speichersektoren mit Daten von dem Host-Prozessor beschrieben werden,

dadurch gekennzeichnet, daß wenigstens ein zusätzlicher Sektorenblock vorgesehen ist, der modifizierte Versionen einzelner Sektoren eines Sektorzuordnungs-Tabellenblocks enthält, und wobei, wenn alle Sektoren in einem solchen Zusatzblock mit modifizierten Versionen von Sektorzuordnungs-Tabellensektor(en) beschrieben sind, der zugehörige Sektorzuordnungs-Tabellenblock neu beschrieben wird, um alle modifizierten Versionen in dem Zusatzblock einzuschließen, und der Zusatzblock gelöscht wird.

2. Speichersystem nach Anspruch 1, wobei der Schreibezeiger durch die Steuereinheit dazu gesteuert wird, sich in einer vorbestimmten Reihenfolge durch die Blöcke zu bewegen, die als gelöscht behandelt werden.

3. Speichersystem nach Anspruch 1 oder 2, wobei die physikalischen Sektoradressen in der Sektorzuordnungstabelle durch logische Sektoradressen geordnet sind, wodurch der N-te Sektorzuordnungstabelleneintrag die physikalische Adresse eines Sektors enthält, in den die Daten mit der logischen Adresse N geschrieben wurden.

4. Speichersystem nach Anspruch 3, wobei die Steuereinheit so ausgelegt ist, daß bei Empfang eines Sektorlesebefehls von dem Host-Prozessor die Steuereinheit eine von dem Host-Prozessor empfangene logische Sektoradresse in der Sektorzuordnungstabelle nachschlägt, um die physikalische Sektoradresse zu erhalten, die die Steuereinheit der logischen Sektoradresse zuvor zugeordnet hat.

5. Speichersystem nach einem der vorstehenden Ansprüche, wobei die Sektorzuordnungstabelle in wenigstens einem der Speichersektorenblöcke in dem Festkörperspeicher gespeichert ist.

6. Speichersystem nach Anspruch 5, wobei die Steuereinheit dazu ausgelegt ist, die Sektorzuordnungstabelle durch Überschreiben der Sektorzuordnungstabelle in ganzen Blöcken zu aktualisieren.

7. Speichersystem nach Anspruch 1, wobei jeder Sektor in dem Zusatzblock die physikalische Adresse

12/21/2022 19:59:42 Page 26 of 57
SAMSUNG EX1002, Page 240 of 1031

des Sektors in dem Sektorzuordnungs-Tabellenblock, den er aktualisiert, und die modifizierte Version des Sektors in dem Sektorzuordnungs-Tabellenblock enthält.

8. Speichersystem nach einem der vorstehenden Ansprüche, wobei die Steuereinheit dazu ausgelegt ist, den Schreibezeiger so zu steuern, daß er sich sequentiell in aufsteigender numerischer Reihenfolge der physikalischen Adressen durch die gelöschten Blöcke bewegt, wenn jeder Block mit darin geschriebenen Daten gefüllt wird.

9. Speichersystem nach Anspruch 8, wobei die Steuerung des Schreibezeigers zyklisch erfolgt, in dem Sinne, daß sobald die Sektoren in dem gemäß der physikalischen Adressenreihenfolge höchsten Block mit Daten gefüllt wurden, der Schreibezeiger durch die Steuereinheit dazu gesteuert wird, zum Sektorenblock mit der numerisch niedrigsten physikalischen Blockadresse von allen gegenwärtig durch die Steuereinheit als gelöscht behandelten Blöcken umzuspringen.

10. Speichersystem nach einem der Ansprüche 1 bis 7, wobei die Steuereinheit dazu ausgelegt ist, eine nicht sequentielle Bewegung des Schreibezeigers gemäß der physikalischen Adressreihenfolge durch die gelöschten Blöcke zu steuern.

11. Speichersystem nach einem der Ansprüche 1 bis 10, wobei jeder Speichersektor physikalisch in einen Datenbereich (300) und einen Reservebereich (302) unterteilt ist, und die Steuereinheit dazu ausgelegt ist, Organisationsdaten, die Kopfdaten und Fehlerkorrekturcode-Daten enthalten, an einer Position in den Sektor zu schreiben, die vom Beginn des Datenbereichs des Sektors versetzt ist, und von dem Host-Prozessor empfangene Benutzerdaten in den im Sektor verbleibenden Bereich auf einer der beiden Seiten der Organisationsdaten zu schreiben.

12. Speichersystem nach Anspruch 11, wobei die Organisationsdaten um einen Betrag versetzt sind, der durch wenigstens ein Bit der in den Sektor zu schreibenden Benutzerdaten bestimmt ist.

13. Speichersystem nach einem der vorstehenden Ansprüche, wobei die Speichersektoren in jedem Sektorenblock zusammen als Einheit löschbar sind.

14. Speichersystem nach Anspruch 13, wobei die Speichersektoren in jedem Sektorenblock auch einzeln löschbar sind.

15. Speichersystem nach einem der vorstehenden Ansprüche, wobei die Steuereinheit dazu ausgelegt ist, Löschoperationen in dem Speicher so zu steuern, daß nur ganze Blöcke von Speichersektoren gelöscht werden, und wobei ein Sektorenblock durch die Steuereinheit als gelöscht behandelt wird, wenn alle Speichersektoren darin gelöschte Sektoren sind.

16. Speichersystem nach Anspruch 15, wobei, falls ein Block einen oder mehrere fehlerhafte Sektoren enthält, die Steuereinheit den gesamten Block als fehlerhaft definiert und diesen Block als einen nicht gelöschten Block behandelt, wodurch keine Daten darin geschrieben werden.

17. Speichersystem nach Anspruch 14, wobei, falls ein Block einen oder mehrere fehlerhafte Sektoren enthält, die Steuereinheit diesen Block als einen gelöschten Block behandelt, wodurch die Steuereinheit immer noch fehlerfreie Sektoren in dem Block zum Speichern von Daten verwenden kann, und wobei das Speichersystem eine Tabelle zur Identifikation fehlerhafter Sektoren enthält, und die Steuereinheit dazu ausgelegt ist, zu prüfen, ob die nächste Sektorenadresse, zu der der Schreibezeiger zu bewegen ist, die Adresse eines fehlerhaften Sektors ist, und, falls es die Adresse eines fehlerhaften Sektors ist, den Schreibezeiger so zu steuern, daß er diesen fehlerhaften Sektor überspringt und sich zur nächsten Sektoradresse gemäß der vorbestimmten Reihenfolge, in der die Sektoren zu beschreiben sind, bewegt.

18. Speichersystem nach einem der vorstehenden Ansprüche, wobei jeder Sektorenblock eine physikalische Blockadresse aufweist, die seine physikalische Position in dem Speicher definiert, und die physikalische Adresse jedes Speichersektors die physikalische Blockadresse des Blocks aufweist, in dem er angeordnet ist, und wobei die Steuereinheit dazu ausgelegt ist, eine Liste der physikalischen Blockadressen wenigstens einiger der als gelöscht behandelten Sektorenblöcke zusammenzustellen, die in einer Reihenfolge, in der sich der Schreibezeiger durch die Blöcke bewegen soll, aufgelistet sind, wobei die Liste durch die Steuereinheit genutzt wird, um schnell den nächsten Sektorenblock zu identifizieren, in den geschrieben wird, und das Speichersystem ferner eine temporäre Speichereinrichtung aufweist, in die die Liste durch die Steuereinheit zu speichern ist.

19. Speichersystem nach einem der vorstehenden Ansprüche, wobei die Steuereinheit dazu ausgelegt ist, daß, wenn ein Sektorschreibebefehl durch die Steuereinheit von dem Host-Prozessor empfangen wird, der veraltete Daten ergibt, die zuvor in einen anderen Sektor geschrieben wurden, die Steuereinheit die Adresse des Sektors mit den nun veral-

12/21/2022 19:59:42 Page 27 of 57
SAMSUNG EX1002, Page 241 of 1031

teten Daten in einem temporären Speicher des Speichersystems speichert.

20. Speichersystem nach Anspruch 19, wobei die Steuereinheit ferner dazu ausgelegt ist, falls ein von einem Benutzer erzeugter Sektorlöschbefehl von dem Host-Prozessor durch die Steuereinheit empfangen wird, die Steuereinheit den zu löschenden Sektor als veraltet markiert und die Adresse des Sektors in dem temporären Speicher speichert.

21. Speichersystem nach Anspruch 19 oder 20, wobei die Steuereinheit dazu ausgelegt ist, nur einer fest vorgegebenen Anzahl von Blöcken zur gleichen Zeit, im folgenden als gegenwärtig veraltete Blöcke bezeichnet, erlaubt, einen oder mehrere Sektoren mit veralteten Daten zu enthalten, die durch den Schreibezeiger geschrieben wurden, und wenn alle Sektoren in dem gegenwärtig veralteten Block veraltete Daten enthalten, der gegenwärtig veraltete Block unverzüglich gelöscht wird.

22. Speichersystem nach Anspruch 21, wobei die Steuereinheit dazu ausgelegt ist, wenn ein Sektor in einem anderem als dem gegenwärtig veralteten Block veraltete Daten enthalten soll: jegliche Daten in gültigen (nicht veralteten) Sektoren in einem solchen gegenwärtig veralteten Block einem anderen Block neu zuzuordnen und dann den gegenwärtig veralteten Block zu löschen; den Sektor in dem anderem als dem gegenwärtig veralteten Block als veraltet zu markieren; und den Block als neuen, gegenwärtig veralteten Block zu bezeichnen.

23. Speichersystem nach Anspruch 21 oder 22, wobei die feste vorbestimmte Anzahl von gegenwärtig veralteten Blöcken eins ist.

24. Speichersystem nach Anspruch 22, wobei der Block, in den die Steuereinheit die gültigen Daten neu zuordnet, der Block ist, in dem der Schreibezeiger gegenwärtig angeordnet ist.

25. Speichersystem nach Anspruch 22, wobei das Speichersystem ferner einen weiteren Schreibezeiger aufweist, im folgenden als Neuzuordnungszeiger bezeichnet, um zu der physikalischen Adresse des Sektors zu zeigen, dem die gültigen Daten neu zugeordnet werden sollen, wobei der Neuzuordnungszeiger immer in einem anderen Sektorenblock als der Schreibezeiger ist.

26. Speichersystem nach Anspruch 25, wobei das Speichersystem ferner einen weiteren Schreibezeiger, im folgenden als Systemschreibezeiger bezeichnet, aufweist, der zu der physikalischen Adresse des Sektors zeigt, in den Systemschreibedaten von dem Host geschrieben werden sollen, wobei der Systemschreibezeiger immer in einem anderen Block als der Schreibezeiger ist.

27. Speichersystem nach Anspruch 26, wobei die Steuereinheit dazu ausgelegt ist, die Existenz wenigstens zweier Blöcke mit einem oder mehreren veralteten Sektoren gleichzeitig zu erlauben, wobei einer den gegenwärtig veralteten Block und der andere einen gegenwärtig veralteten Systemblock mit einem oder mehreren veralteten Systemdatensektoren darstellt, und falls einer der Systemdatensektoren neu zuzuordnen ist, um das Löschen des gegenwärtigen veralteten Systemblocks zu ermöglichen, die neu zugeordneten Systemdaten zu der Adresse gesendet werden, auf die der Systemschreibezeiger gegenwärtig zeigt.

28. Speichersystem nach Anspruch 26, wobei das Speichersystem einen weiteren Schreibezeiger aufweist, im folgenden Systemneuzuordnungszeiger bezeichnet, um zu der physikalischen Adresse des Sektors zu zeigen, zu der gültige Systemdaten neu zugeordnet werden sollen, wobei der Systemneuzuordnungszeiger immer in einem anderen Sektorenblock als der Schreibezeiger und der Systemschreibezeiger ist.

29. Speichersystem nach einem der Ansprüche 26 bis 28, wobei die Steuereinheit dazu ausgelegt ist, falls der gegenwärtig veraltete Block zu der Zeit, in der die Steuereinheit einen solchen gegenwärtig veralteten Block löschen muß, weil veraltete Daten gerade in einem anderen Block erzeugt wurden, einen der Schreibezeiger enthält, mit der Erzeugung eines neuen gegenwärtig veralteten Blocks fortzufahren, aber das Löschen des alten gegenwärtig veralteten Blocks, im folgenden als bestehender veralteter Block bezeichnet, aufzuschieben bis alle gelöschten Sektoren in dem bestehenden veralteten Block gefüllt sind und der Zeiger sich, wie durch die Steuereinheit definiert, zu dem nächsten gelöschten, zu verwendenden Block bewegt, und gleichzeitig alle gültigen (nicht veralteten) Daten in den bestehenden veralteten Block durch die Steuereinheit neu zugeordnet werden und der bestehende veraltete Block gelöscht wird.

30. Speichersystem nach Anspruch 27, wobei die Steuereinheit dazu ausgelegt ist, in einem temporären Speicher des Speichersystems zugehörige Listen von logischen Sektoradressen zu speichern, die Sektoren in dem Speicher entsprechen, in die neu zugeordnete Daten durch den Neuzuordnungszeiger, im folgenden als Neuzuordnungssektorenliste bezeichnet, den Systemschreibezeiger, im folgenden als Schreibsystemsektorenliste bezeichnet, und den Systemneuzuordnungszeiger, im folgenden als Systemneuzuordnungssektorenliste nach

letzter Aktualisierung der Sektorenzuordnungstabelle bezeichnet, geschrieben wurden, und die Steuereinheit dazu ausgelegt ist, in dem temporären Speicher entsprechende Listen der Reihenfolge von Blöcken, die durch den Neuzuordnungszeiger, den Systemschreibezeiger und den Systemneuzuordnungszeiger verwendet wurden, zu speichern, im folgenden als Neuzuordnungsblockliste, Schreibsystemblockliste und Systemneuzuordnungsblockliste (SRBL) bezeichnet.

31. Speichersystem nach einem der vorstehenden Ansprüche, wobei zusätzlich zum Schreiben von Datenstrukturen von dem Host-Prozessor in den Speicher die Steuereinheit auch als Steuerinformationen **gekennzeichnete** Daten erzeugt und in den Speicher schreibt, und die Steuereinheit dazu ausgelegt ist, solche Steuerinformationen in einem oder mehreren unterschiedlichen Steuerblöcken der Blöcke von Speichersektoren in jene Blöcke zu schreiben, in die von dem Host-Prozessor empfangene Datenstrukturen geschrieben sind.

32. Speichersystem nach Anspruch 31, wobei die Steuereinheit dazu ausgelegt ist, in wenigstens einem der Steuerblöcke eine Liste der Blockadressen aller Sektorzuordnungs-Tabellenblöcke zu speichern.

33. Speichersystem nach Anspruch 31 oder 32, wobei die Steuereinheit dazu ausgelegt ist, die Blockadressen des mindestens einen Steuerblocks in einem zugewiesenen Boot-Block des Speichers zu speichern, wobei dieser zugewiesene Boot-Block der erste Block des Sektors in dem Speicher ist, der keinen fehlerhaften Sektor enthält.

34. Speichersystem nach Anspruch 32, wenn abhängig von Anspruch 1 oder 7, wobei die Liste aller Sektorzuordnungs-Tabellenblockadressen in Form mehrerer Listenabschnitte vorliegt und jeder Abschnitt die Blockadressen einer Gruppe logisch fortlaufender Sektorzuordnungs-Tabellenblöcke und jeglicher zugehöriger Zusatzsektorzuordnungs-Tabellenblöcke enthält.

35. Speichersystem nach einem der Ansprüche 1 und 8, wobei die Steuereinheit dazu ausgelegt ist, in einen temporären Speicher des Speichersystems eine Schreibsektorliste von logischen Sektoradressen für Datenstrukturen zu speichern, die durch den Schreibezeiger seit der letzten Aktualisierung des SAT geschrieben wurden.

36. Speichersystem nach Anspruch 35, wobei die Steuereinheit dazu ausgelegt ist, in dem temporären Speicher auch die Reihenfolge zu speichern, in der die Blöcke durch den Schreibezeiger zum Schreiben der Daten seit der letzten Aktualisierung der Sektorzuordnungstabelle verwendet wurden, wobei die Reihenfolge in Form einer Schreibblockliste der Blockadressen der Blöcke gespeichert ist, in der die aktualisierten Sektoren angeordnet sind, deren Adressen in der Schreibsektorliste abgelegt sind.

37. Speicher nach Anspruch 36, wobei die Schreibsektorliste eine vorbestimmte Größe aufweist, und sobald die Schreibsektorliste voll ist, wenigstens ein Sektorzuordnungs-Tabellenblock oder ein Zusatzsektorzuordnungs-Tabellenblock aktualisiert wird und die Schreibsektorliste und die Schreibblockliste geleert werden.

38. Speichersystem nach Anspruch 36, wobei die Steuereinheit eine physikalische Startsektoradresse und die Verbindungen zwischen Blöcken, die Sektoren enthalten, in die Daten durch die Steuereinheit seit der letzten Aktualisierung einer Sektorzuordnungstabelle oder eines Zusatzsektorzuordnungs-Tabellenblocks geschrieben wurden, in einen solchen Steuerblock des Festkörperspeichers speichert.

39. Speichersystem nach einem der vorstehenden Ansprüche, wobei jeder Sektor aus einer einzelnen Speicher-"Seite" besteht, nämlich einer Reihe von Speicherzellen in einem solchen Block von Speichersektoren.

40. Speichersystem nach einem der vorstehenden Ansprüche, wobei die Steuereinheit dazu ausgelegt ist, Daten in die Speichersektoren in gleichmäßig bemessenen Datensegmenten zu schreiben und aus diesen zu lesen.

41. Speichersystem nach Anspruch 40, wobei alle Speichersektoren die gleiche Größe aufweisen und jedes Datensegment genauso groß ist wie ein solcher Speichersektor.

42. Speichersystem nach einem der vorstehenden Ansprüche, ferner mit einem temporären Cache-Speicher, in dem die Steuereinheit dazu ausgelegt ist, eine Gruppe von fortlaufenden Sektorzuordnungs-Tabelleneinträgen einschließlich der Sektorzuordnungs-Tabelleneinträge zu speichern, auf die von der Sektor-Zuordnungstabelle durch die Steuereinheit als letztes zugegriffen wurde.

43. Speichersystem nach Anspruch 41, wenn abhängig von Anspruch 7, wobei die Steuereinheit dazu ausgelegt ist, in dem temporären Cache-Speicher eine Liste von physikalischen Adressen aller Zusatzsektorzuordnungs-Tabellenblöcke und der Sektorzuordnungs-Tabellenblöcke, die mit diesen verbunden sind, zu erzeugen, die jedesmal aktualisiert

wird, wenn eine Sektorzuordnungs-Tabellensektorschreibeoperation durchgeführt wird.

44. Speichersystem nach einem der vorstehenden Ansprüche, wobei der Festkörperspeicher einen einzelnen Speicherarray in Form eines einzelnen Speicherchips aufweist.

45. Speichersystem nach einem der Ansprüche 1 bis 43, wobei der Festkörperspeicher einen durch mehrere Speicherchips gebildeten Speicherarray aufweist.

46. Speichersystem nach einem der Ansprüche 1 bis 43, wobei der Festkörperspeicher mehrere Speicherarrays in Form mehrerer Speicherchips aufweist, und die Steuereinheit dazu ausgelegt ist, die Speichersektoren in den mehreren Speicherchips in einer Mehrzahl von virtuellen Blöcken zu bilden, wobei jeder virtuelle Block einen löschbaren Block von Speichersektoren von jedem Speicherchip aufweist, und die virtuellen Blöcke in solche zu sortieren, die als gelöscht behandelt werden, und solche, die als nicht gelöscht behandelt werden.

47. Speichersystem nach Anspruch 46, wobei die Steuereinheit dazu ausgelegt ist, eine Liste von virtuellen Blöcken zusammenzustellen, die als gelöscht behandelt werden, und diese in dem temporären Speicher in dem Speichersystem zu speichern, und den Schreibezeiger zu steuern, so daß er sich von einem Chip zum anderen für jede aufeinanderfolgende Sektorschreibeoperation bewegt, beginnend an einem Sektor in einem löschbaren Block des virtuellen Blocks und sich darauffolgend zu einem Sektor in jedem der anderen löschbaren Blöcke in dem virtuellen Block bewegend, bis ein Sektor in jedem löschbaren Block des virtuellen Blocks beschrieben ist, und sich dann zurück zu dem Chip bewegend, in dem der erste Sektor beschrieben wurde und in ähnlicher Weise fortfahrend, um einen anderen Sektor in jedem löschbaren Block des virtuellen Blocks zu füllen, usw. bis der virtuelle Block voll mit Daten ist, und sich dann der Schreibezeiger zu dem nächsten virtuellen Block in der Liste von virtuellen Blöcken, die als gelöscht behandelt werden, bewegt, und diesen nächsten virtuellen Block auf ähnliche Weise füllt.

48. Speichersystem nach Anspruch 47, wobei die Steuereinheit dazu ausgelegt ist, daß sie für alle n aufeinanderfolgenden Sektorschreibeoperationen einen Mehrfach-Sektorschreibebefehl ausführt, der vom Host-Prozessor empfangen wurde, wobei n kleiner oder gleich der Anzahl von Festkörper-Speicherchips in dem Speichersystem ist, wobei die Steuereinheit im wesentlichen gleichzeitig in einen Sektor in jedem der n Chips schreibt.

49. Speichersystem nach Anspruch 47 oder 48, wobei die Steuereinheit dazu ausgelegt ist, die Löschung irgendeines der virtuellen Blöcke durch gleichzeitiges Löschen aller löschbaren Blöcke in dem virtuellen Block auszuführen.

50. Verfahren zum Steuern des Lesens und Schreibens von Datenstrukturen aus einem bzw. in einen Festkörperspeicher (5) mit nicht-flüchtigen Speichersektoren (1, 2, 3), die einzeln adressierbar und in löschbaren Sektorenblöcken (4) angeordnet sind, wobei jeder Sektor eine physikalische Adresse aufweist, die seine physikalische Position in dem Speicher definiert, und in dem Verfahren:

die Sektorenblöcke in Blöcke, die als gelöscht behandelt werden, und Blöcke, die als nicht gelöscht behandelt werden, sortiert werden;
ein Schreibezeiger vorgesehen wird, der auf die physikalische Adresse eines Sektors zeigt, in den die Daten von einem Host-Prozessor zu schreiben sind, wobei der Schreibezeiger so gesteuert wird, daß er sich in einer vorbestimmten Reihenfolge durch die physikalischen Adressen der Speichersektoren eines Blocks bewegt, der als gelöscht behandelt wird, und sich, wenn der Block gefüllt wurde, zu einem anderen gelöschten Block bewegt, und wenn ein Sektor-Schreibebefehl von dem Host-Prozessor empfangen wird, eine von dem Host-Prozessor empfangene logische Adresse in eine physikalische Adresse übersetzt wird, in die Daten geschrieben werden, indem der logischen Adresse die physikalische Adresse zugeordnet wird, auf die der Schreibezeiger gegenwärtig zeigt;
eine Sektorzuordnungstabelle von logischen Adressen der Sektoren mit den zugehörigen physikalischen Adressen der Sektoren, die ihnen durch eine Steuereinheit (8) zugeordnet wurden, in dem nicht-flüchtigen Festkörperspeicher gespeichert werden; und
die Sektorzuordnungstabelle weniger häufig aktualisiert wird als Speichersektoren mit Daten von dem Host-Prozessor beschrieben werden;

dadurch gekennzeichnet, daß wenigstens ein zusätzlicher Sektorenblock vorgesehen ist, der modifizierte Versionen einzelner Sektoren eines Sektorzuordnungs-Tabellenblocks enthält, und wobei, wenn alle Sektoren in einem solchen Zusatzblock mit modifizierten Versionen von Sektorzuordnungs-Tabellensektor(en) beschrieben sind, der zugehörige Sektorzuordnungs-Tabellenblock neu beschrieben wird, um alle modifizierten Versionen in dem Zusatzblock einzuschließen, und der Zusatzblock gelöscht wird.

12/21/2022 19:59:42 Page 30 of 57
SAMSUNG EX1002, Page 244 of 1031

Revendications

1. Système de mémoire (5, 8) pour une connexion à un processeur hôte, le système comprenant :

une mémoire à semi-conducteurs (5) comportant des secteurs de mémoire non volatile (1, 2, 3) qui sont adressables individuellement et qui sont agencés en blocs effaçables (4) de secteurs, chacun desdits secteurs ayant une adresse physique définissant sa position physique dans la mémoire ;
et un contrôleur (8) pour écrire des structures de données et lire des structures de données dans la mémoire et pour trier les blocs de secteurs en blocs qui sont traités comme étant effacés et en blocs qui sont traités comme n'étant pas effacés ; dans lequel le contrôleur comprend :

des moyens (11) pour transformer des adresses logiques reçues du processeur hôte en adresses physiques desdits secteurs de mémoire dans la mémoire ;
un pointeur d'écriture pour pointer vers l'adresse physique d'un secteur dans lequel des données provenant du processeur hôte doivent être écrites, ledit pointeur d'écriture étant commandé par le contrôleur pour se déplacer dans un ordre prédéterminé parmi les adresses physiques des secteurs de mémoire de n'importe quel bloc qui est traité comme étant effacé et, lorsque le bloc a été rempli, pour se déplacer vers un autre des blocs effacés ;
et dans lequel le contrôleur est configuré de sorte que, lorsqu'une commande d'écriture de secteur est reçue du processeur hôte, le contrôleur transforme une adresse logique reçue du processeur hôte en une adresse physique à laquelle des données sont écrites en attribuant pour ladite adresse logique cette adresse physique vers laquelle ledit pointeur d'écriture pointe actuellement, et dans lequel le contrôleur est configuré pour compiler une table d'attribution de secteurs d'adresses logiques de secteurs avec des adresses physiques de secteurs respectives qui ont été attribuées à cette fin par le contrôleur, et dans lequel le contrôleur met à jour la table d'attribution de secteurs moins fréquemment qu'il n'y a d'écriture de données provenant du processeur hôte dans des secteurs de mémoire, **caractérisé en ce qu'**il est prévu au moins un bloc supplémentaire de secteurs contenant des versions modifiées de secteurs individuels d'un bloc de table d'attribution de secteurs, et dans lequel, lorsque tous les secteurs dans un dit bloc supplémentaire sont écrits avec des versions modifiées de secteur(s) de table d'attribution de secteurs, le bloc de table d'attribution de secteurs respectif est réécrit de manière à inclure toutes les versions modifiées dans le bloc supplémentaire et le bloc supplémentaire est effacé.

2. Système de mémoire selon la revendication 1, dans lequel ledit pointeur d'écriture est commandé par le contrôleur pour se déplacer dans un ordre prédéterminé parmi les blocs qui sont traités comme étant effacés.

3. Système de mémoire selon la revendication 1 ou la revendication 2, dans lequel les adresses physiques de secteurs dans la table d'attribution de secteurs sont ordonnées par adresses logiques de secteurs, moyennant quoi la Nième entrée de table d'attribution de secteurs contient l'adresse physique d'un secteur dans lequel des données ayant une adresse logique N ont été écrites.

4. Système de mémoire selon la revendication 3, dans lequel le contrôleur est configuré de sorte que, lorsqu'une commande de lecture de secteur est reçue du processeur hôte, le contrôleur consulte une adresse logique de secteur reçue du processeur hôte dans la table d'attribution de secteurs afin d'obtenir l'adresse physique de secteur que le contrôleur a attribuée précédemment à ladite adresse logique de secteur.

5. Système de mémoire selon l'une quelconque des revendications précédentes, dans lequel la table d'attribution de secteurs est mémorisée dans au moins l'un desdits blocs de secteurs de mémoire dans la mémoire à semi-conducteurs.

6. Système de mémoire selon la revendication 5, dans lequel le contrôleur est configuré pour mettre à jour la table d'attribution de secteurs en réécrivant la table d'attribution de secteurs par blocs entiers.

7. Système de mémoire selon la revendication 1, dans lequel chaque secteur dans un dit bloc supplémentaire contient l'adresse physique du secteur du bloc de table d'attribution de secteurs qu'il met à jour et la version modifiée dudit secteur du bloc de table d'attribution de secteurs.

8. Système de mémoire selon l'une quelconque des revendications précédentes, dans lequel le contrôleur est configuré pour commander le pointeur d'écriture de manière à ce qu'il se déplace de manière séquentielle, dans l'ordre numérique ascen-

12/21/2022 19:59:42 Page 31 of 57
SAMSUNG EX1002, Page 245 of 1031

dant des adresses physiques, parmi les blocs effacés, alors que chaque bloc est rempli de données écrites dans celui-ci.

9. Système de mémoire selon la revendication 8, dans lequel la commande du pointeur d'écriture est cyclique dans le sens où une fois que les secteurs dans le bloc le plus élevé, selon l'ordre des adresses physiques, ont été remplis de données, le pointeur d'écriture est commandé par le contrôleur pour retourner au bloc de secteurs ayant l'adresse physique de bloc la plus faible numériquement parmi tous les blocs actuellement traités par le contrôleur comme étant effacés.

10. Système de mémoire selon l'une quelconque des revendications 1 à 7, dans lequel le contrôleur est configuré pour commander le pointeur d'écriture de manière à ce qu'il se déplace de manière non séquentielle, selon l'ordre des adresses physiques, parmi les blocs effacés.

11. Système de mémoire selon l'une quelconque des revendications 1 à 10, dans lequel chacun desdits secteurs de mémoire est divisé physiquement en une zone de données (300) et une zone de réserve (302) et le contrôleur est configuré pour écrire des données supplémentaires comprenant des données d'en-tête et des données de code de correction d'erreurs à une position dans le secteur qui est décalée du début de la zone de données du secteur et pour écriture des données d'utilisateur, reçues du processeur hôte, dans l'espace restant dans le secteur, de chaque côté des données supplémentaires.

12. Système de mémoire selon la revendication 11, dans lequel lesdites données supplémentaires sont décalées d'une quantité qui est déterminée par au moins un bit des données d'utilisateur devant être écrites dans le secteur.

13. Système de mémoire selon l'une quelconque des revendications précédentes, dans lequel les secteurs de mémoire dans chacun desdits blocs de secteurs sont effaçables ensemble en tant qu'unité.

14. Système de mémoire selon la revendication 13, dans lequel les secteurs de mémoire dans chacun desdits blocs de secteurs sont également effaçables individuellement.

15. Système de mémoire selon l'une quelconque des revendications précédentes, dans lequel le contrôleur est configuré pour commander des opérations d'effacement sur la mémoire de manière à n'effacer que des blocs entiers de secteurs de mémoire, et dans lequel un bloc de secteurs est traité par le contrôleur comme un bloc effacé si tous les secteurs de mémoire dans celui-ci sont des secteurs effacés.

16. Système de mémoire selon la revendication 15, dans lequel, si un bloc contient un ou plusieurs mauvais secteurs, le contrôleur définit le bloc entier comme étant mauvais et traite ce bloc comme un bloc non effacé, moyennant quoi aucune donnée ne sera écrite dans celui-ci.

17. Système de mémoire selon la revendication 14, dans lequel, si un bloc contient un ou plusieurs mauvais secteurs, le contrôleur traite ce bloc comme un bloc effacé, moyennant quoi le contrôleur peut encore utiliser de bons secteurs dans le bloc pour mémoriser des données, et dans lequel le système de mémoire comprend une table identifiant de mauvais secteurs et le contrôleur est configuré pour vérifier si l'adresse de secteur suivante vers laquelle le pointeur d'écriture doit être déplacé est l'adresse d'un mauvais secteur et, s'il s'agit de l'adresse d'un mauvais secteur, pour commander le pointeur d'écriture pour qu'il saute ce mauvais secteur et se déplace vers l'adresse de secteur suivante selon l'ordre prédéterminé dans lequel les secteurs doivent être écrits.

18. Système de mémoire selon l'une quelconque des revendications précédentes, dans lequel chaque bloc de secteurs a une adresse physique de bloc définissant sa position physique dans la mémoire et l'adresse physique de chacun desdits secteurs de mémoire comprend l'adresse physique de bloc du bloc dans lequel il est situé, et dans lequel le contrôleur est configuré pour compiler une liste des adresses physiques de blocs d'au moins certains des blocs de secteurs traités comme étant effacés, énumérées dans un ordre dans lequel le pointeur d'écriture doit se déplacer parmi les blocs, laquelle liste est utilisée par le contrôleur afin d'identifier rapidement le bloc suivant de secteurs devant être écrits, et le système de mémoire comprend, en outre, des moyens formant mémoire temporaire dans lesquels ladite liste est mémorisée par le contrôleur.

19. Système de mémoire selon l'une quelconque des revendications précédentes, dans lequel le contrôleur est configuré de sorte que, lorsqu'une commande d'écriture de secteur est reçue du processeur hôte par le contrôleur, laquelle commande rend obsolètes des données écrites précédemment dans un autre secteur, le contrôleur mémorise dans une mémoire temporaire du système de mémoire l'adresse du secteur contenant les données maintenant obsolètes.

20. Système de mémoire selon la revendication 19,

dans lequel le contrôleur est, en outre, configuré de sorte que, si une commande d'effacement de secteur, générée par un utilisateur, est reçue du processeur hôte par le contrôleur, le contrôleur marque comme obsolète le secteur devant être effacé et mémorise l'adresse du secteur dans ladite mémoire temporaire.

21. Système de mémoire selon la revendication 19 ou la revendication 20, dans lequel le contrôleur est configuré pour ne permettre qu'à un nombre prédéterminé fixé de blocs à tout moment, appelés ici les blocs obsolètes actuels, de contenir un ou plusieurs secteurs contenant des données obsolètes qui ont été écrites par le pointeur d'écriture, et de sorte que, lorsque tous les secteurs dans l'un desdits blocs obsolètes actuels contiennent des données obsolètes, ledit bloc obsolète actuel soit immédiatement effacé.

22. Système de mémoire selon la revendication 21, dans lequel le contrôleur est configuré de sorte que, lorsqu'un secteur dans un bloc autre qu'un desdits blocs obsolètes actuels contient des données obsolètes, le contrôleur déplace toutes données dans des secteurs valides (non obsolètes) dans l'un desdits blocs obsolètes actuels dans un autre bloc et efface ensuite ledit bloc obsolète actuel, marque ledit secteur dans ledit bloc autre qu'un bloc obsolète actuel comme étant obsolète et désigne ledit autre bloc en tant que nouveau bloc obsolète actuel.

23. Système de mémoire selon la revendication 21, ou la revendication 22, dans lequel ledit nombre prédéterminé fixé de blocs obsolètes actuels est un.

24. Système de mémoire selon la revendication 22, dans lequel ledit bloc dans lequel le contrôleur déplace lesdites données valides est le bloc dans lequel le pointeur d'écriture est actuellement situé.

25. Système de mémoire selon la revendication 22, dans lequel le système de mémoire comprend un autre pointeur d'écriture, appelé ici pointeur de déplacement, pour pointer vers l'adresse physique du secteur dans lequel lesdites données valides doivent être déplacées, le pointeur de déplacement étant toujours dans un bloc différent de secteurs par rapport au pointeur d'écriture.

26. Système de mémoire selon la revendication 25, dans lequel le système de mémoire comprend un autre pointeur d'écriture, appelé pointeur d'écriture de système, qui pointe vers l'adresse physique du secteur dans lequel des données de système doivent être écrites par l'hôte, le pointeur d'écriture de système étant toujours dans un bloc différent par rapport au pointeur d'écriture.

27. Système de mémoire selon la revendication 26, dans lequel le contrôleur est configuré pour permettre l'existence d'au moins deux blocs qui contiennent un ou plusieurs secteurs obsolètes à tout moment, l'un étant ledit bloc obsolète actuel et l'autre étant un bloc de système obsolète actuel contenant un ou plusieurs secteurs de données de système obsolètes et, si des secteurs de données de système doivent être déplacés afin de permettre l'effacement du bloc de système obsolète actuel, les données de système déplacées sont envoyées à l'adresse vers laquelle le pointeur d'écriture de système pointe actuellement.

28. Système de mémoire selon la revendication 26, dans lequel le système de mémoire comprend un autre pointeur d'écriture, appelé ici pointeur de déplacement de système, pour pointer vers l'adresse physique du secteur dans lequel des données de système valides doivent être déplacées, le pointeur de déplacement de système étant toujours dans un bloc différent de secteurs par rapport au pointeur d'écriture et au pointeur d'écriture de système.

29. Système de mémoire selon l'une quelconque des revendications 26 à 28, dans lequel le contrôleur est configuré de sorte que, si le bloc obsolète actuel contient l'un desdits pointeurs d'écriture au moment où le contrôleur doit effacer l'un desdits blocs obsolètes actuels parce que des données obsolètes viennent d'être créées dans un autre bloc, le contrôleur procède à la création d'un nouveau bloc obsolète actuel mais reporte l'effacement de l'ancien bloc obsolète actuel, appelé ici le bloc obsolète en attente, jusqu'à ce que tous les secteurs effacés dans le bloc obsolète en attente aient été remplis et que ledit pointeur se déplace vers le bloc effacé suivant devant être utilisé, tel que défini par le contrôleur, moment auquel toutes données valides (non obsolètes) dans le bloc obsolète en attente sont déplacées par le contrôleur et le bloc obsolète en attente est effacé.

30. Système de mémoire selon la revendication 27, dans lequel le contrôleur est configuré pour mémoriser dans une mémoire temporaire du système de mémoire des listes respectives d'adresses logiques de secteurs correspondant à des secteurs dans la mémoire dans lesquels des données déplacées ont été écrites par le pointeur de déplacement, appelée ici liste de secteurs de déplacement, par le pointeur d'écriture de système, appelée ici liste de secteurs d'écriture de système, et par le pointeur de déplacement de système, appelée ici liste de secteurs de déplacement de système, depuis la dernière mise à jour de la table d'attribution de secteurs, et le contrôleur est configuré pour mémoriser dans lesdites listes correspondantes de mémoire temporaire de

12/21/2022 19:59:42 Page 33 of 57
SAMSUNG EX1002, Page 247 of 1031

l'ordre des blocs qui ont été utilisés par le pointeur de déplacement, le pointeur d'écriture de système et le pointeur de déplacement de système, appelées ici liste de blocs de déplacement, liste de blocs d'écriture de système et liste de blocs de déplacement de système (SRBL).

31. Système de mémoire selon l'une quelconque des revendications précédentes, dans lequel en plus d'écrire des structures de données dans la mémoire, provenant du processeur hôte, le contrôleur génère et écrit également dans la mémoire des données désignées comme étant des informations de commande, et le contrôleur est configuré pour écrire ces informations de commande dans un ou plusieurs blocs de commande des blocs de secteurs de mémoire différents de ceux dans lesquels des structures de données reçues du processeur hôte sont écrites.

32. Système de mémoire selon la revendication 31, dans lequel le contrôleur est configuré pour mémoriser dans au moins l'un desdits blocs de commande une liste des adresses de blocs de tous les blocs de table d'attribution de secteurs.

33. Système de mémoire selon la revendication 31 ou la revendication 32, dans lequel le contrôleur est configuré pour mémoriser les adresses de blocs desdits un ou plusieurs blocs de commande dans un bloc d'amorçage dédié de la mémoire, ce bloc d'amorçage qui est un bloc dédié étant le premier bloc de secteurs dans la mémoire qui ne contient aucun mauvais secteur.

34. Système de mémoire selon la revendication 32, telle que dépendant de la revendication 1 ou 7, dans lequel ladite liste de toutes les adresses de blocs de table d'attribution de secteurs est sous la forme d'une pluralité de parties de liste et chacune desdites parties contient les adresses de blocs d'un groupe de blocs de table d'attribution de secteurs logiquement contigus et de tous blocs de table d'attribution de secteurs supplémentaires correspondant à ceux-ci.

35. Système de mémoire selon l'une quelconque des revendications 1 et 8, dans lequel le contrôleur est configuré pour mémoriser dans une mémoire temporaire du système de mémoire une liste d'adresses de secteurs logiques de secteurs d'écriture pour des structures de données qui ont été écrites par le pointeur d'écriture depuis la dernière mise à jour de la table d'attribution de secteurs.

36. Système de mémoire selon la revendication 35, dans lequel le contrôleur est configuré pour également mémoriser dans ladite mémoire temporaire l'ordre dans lequel des blocs ont été utilisés par le pointeur d'écriture pour écrire des données depuis la dernière mise à jour de la table d'attribution de secteurs, cet ordre étant mémorisé sous la forme d'une liste d'adresses de blocs de blocs d'écriture des blocs dans lesquels les secteurs mis à jour dont les adresses sont conservées dans la liste de secteurs d'écriture sont situés.

37. Mémoire selon la revendication 36, dans laquelle la liste de secteurs d'écriture a une taille prédéterminée et, une fois que la liste de secteurs d'écriture est pleine, au moins un bloc de table d'attribution de secteurs ou un bloc de table d'attribution de secteurs supplémentaire est mis à jour et la liste de secteurs d'écriture et la liste de blocs d'écriture sont vidées.

38. Système de mémoire selon la revendication 36, dans lequel le contrôleur mémorise, dans l'un desdits blocs de commande de la mémoire à semi-conducteurs, une adresse physique de secteur de début et les liens entre des blocs contenant des secteurs dans lesquels des données ont été écrites par le contrôleur depuis la dernière mise à jour d'une table d'attribution de secteurs ou d'un bloc de table d'attribution de secteurs supplémentaire.

39. Système de mémoire selon l'une quelconque des revendications précédentes, dans lequel chacun desdits secteurs consiste en une seule « page » de mémoire, à savoir une rangée de cellules de mémoire dans l'un desdits blocs de secteurs de mémoire.

40. Système de mémoire selon l'une quelconque des revendications précédentes, dans lequel le contrôleur est configuré pour écrire des données et lire des données dans les secteurs de mémoire par segments de données de tailles uniformes.

41. Système de mémoire selon la revendication 40, dans lequel tous les secteurs de mémoire ont la même taille et la taille de chacun desdits segments de données est égale à la taille d'un desdits secteurs de mémoire.

42. Système de mémoire selon l'une quelconque des revendications précédentes, comprenant en outre une mémoire cache temporaire dans laquelle le contrôleur est configuré pour mémoriser, un groupe d'entrées de table d'attribution de secteurs contiguës comprenant l'entrée de table d'attribution de secteurs à laquelle le contrôleur a accédé en dernier, depuis la table d'attribution de secteurs.

43. Système de mémoire selon la revendication 41, lorsqu'elle dépend de la revendication 7, dans le-

12/21/2022 19:59:42 Page 34 of 57
SAMSUNG EX1002, Page 248 of 1031

quel le contrôleur est configuré pour créer dans ladite mémoire cache temporaire une liste d'adresses physiques de tous les blocs de table d'attribution de secteurs supplémentaires et des blocs de table d'attribution de secteurs auxquels ils sont associés, laquelle est mise à jour à chaque fois qu'une opération d'écriture de secteur de table d'attribution de secteurs est effectuée.

44. Système de mémoire selon l'une quelconque des revendications précédentes, dans lequel la mémoire à semi-conducteurs comprend une seule matrice de mémoire sous la forme d'une seule puce de mémoire.

45. Système de mémoire selon l'une quelconque des revendications 1 à 43, dans lequel la mémoire à semi-conducteurs comprend une matrice de mémoire formée par une pluralité de puces de mémoire.

46. Système de mémoire selon l'une quelconque des revendications 1 à 43, dans lequel la mémoire à semi-conducteurs comprend une pluralité de matrices de mémoire sous la forme d'une pluralité de puces de mémoire, et dans lequel le contrôleur est configuré pour former les secteurs de mémoire dans la pluralité de puces de mémoire en une multiplicité de blocs virtuels, chacun desdits blocs virtuels comprenant un bloc effaçable de secteurs de mémoire de chacune desdites puces de mémoire, et pour trier lesdits blocs virtuels en ceux qui sont traités comme étant effacés et ceux qui sont traités comme n'étant pas effacés.

47. Système de mémoire selon la revendication 46, dans lequel le contrôleur est configuré pour compiler une liste des blocs virtuels traités comme étant effacés et pour mémoriser celle-ci dans une mémoire temporaire dans le système de mémoire, et pour commander le pointeur d'écriture pour qu'il se déplace d'une puce vers une autre pour chaque opération d'écriture de secteur consécutive, en commençant par un secteur dans un bloc effaçable du bloc virtuel et en se déplaçant consécutivement vers un secteur dans chacun des autres blocs effaçables dans le bloc virtuel jusqu'à ce qu'un secteur ait été écrit dans chaque bloc effaçable du bloc virtuel, et en retournant ensuite vers la puce dans laquelle le premier secteur a été écrit et en procédant d'une manière similaire pour remplir un autre secteur dans chaque bloc effaçable du bloc virtuel, et ainsi de suite jusqu'à ce que le bloc virtuel soit plein de données, et pour ensuite déplacer le pointeur d'écriture vers le bloc virtuel suivant dans ladite liste de blocs virtuels traités comme étant effacés, et remplir ce bloc virtuel suivant d'une manière similaire.

48. Système de mémoire selon la revendication 47, dans lequel le contrôleur est configuré de sorte que, pour toutes les n opérations d'écriture de secteurs contigus que le contrôleur exécute pour une commande d'écriture de secteurs multiples reçue du processeur hôte, où n est inférieur ou égal au nombre de puces de mémoire à semi-conducteurs dans le système de mémoire, le contrôleur écrive sensiblement simultanément dans un secteur dans chacune des n puces.

49. Système de mémoire selon la revendication 47 ou la revendication 48, dans lequel le contrôleur est configuré pour effectuer l'effacement de n'importe lequel desdits blocs virtuels en effaçant simultanément tous les blocs effaçables dans le bloc virtuel.

50. Procédé de contrôle de lecture et d'écriture de structures de données dans une mémoire à semi-conducteurs (5) comportant des secteurs (1, 2, 3) de mémoire non volatile qui sont adressables individuellement et qui sont agencés en blocs (4) effaçables de secteurs, chacun desdits secteurs ayant une adresse physique définissant sa position physique dans la mémoire, le procédé comprenant les étapes consistant à :

trier les blocs de secteurs en blocs qui sont traités comme étant effacés et en blocs qui sont traités comme n'étant pas effacés ;
prévoir un pointeur d'écriture pour pointer vers l'adresse physique d'un secteur dans lequel des données provenant d'un processeur hôte doivent être écrites, et commander ledit au moins un pointeur d'écriture pour qu'il se déplace dans un ordre prédéterminé parmi les adresses physiques des secteurs de mémoire de n'importe quel bloc qui est traité comme étant effacé et, lorsque le bloc a été rempli, qu'il se déplace vers un autre des blocs effacés et, lorsqu'une commande d'écriture de secteur est reçue du processeur hôte, transformer une adresse logique reçue du processeur hôte en une adresse physique à laquelle des données sont écrites en attribuant pour ladite adresse logique l'adresse physique vers laquelle ledit pointeur d'écriture pointe actuellement ;
mémoriser dans une mémoire à semi-conducteurs non volatile une table d'attribution de secteurs d'adresses logiques de secteurs avec des adresses physiques respectives de secteurs qui ont été attribuées à cette fin par un contrôleur (8);
et mettre à jour la table d'attribution de secteurs moins fréquemment qu'il n'y a d'écriture de données provenant du processeur hôte dans des secteurs de mémoire ;

12/21/2022 19:59:42 Page 35 of 57
SAMSUNG EX1002, Page 249 of 1031

caractérisé en ce qu'il est prévu au moins un bloc supplémentaire de secteurs contenant des versions modifiées de secteurs individuels d'un bloc de table d'attribution de secteurs, et dans lequel, lorsque tous les secteurs dans un dit bloc supplémentaire sont écrits avec des versions modifiées de secteur(s) de table d'attribution de secteurs, le bloc de table d'attribution de secteurs respectif est réécrit de manière à inclure toutes les versions modifiées dans le bloc supplémentaire et le bloc supplémentaire est effacé.

12/21/2022 19:59:42 Page 36 of 57
SAMSUNG EX1002, Page 250 of 1031

Fig.1.

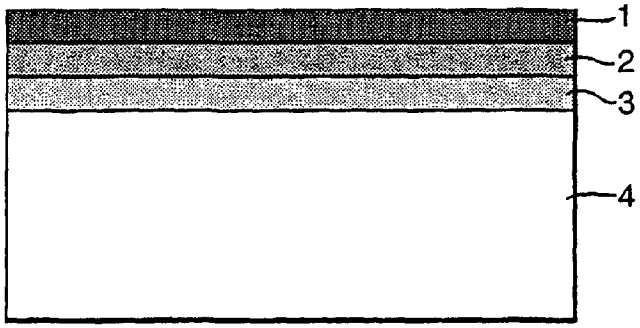


Fig.2.

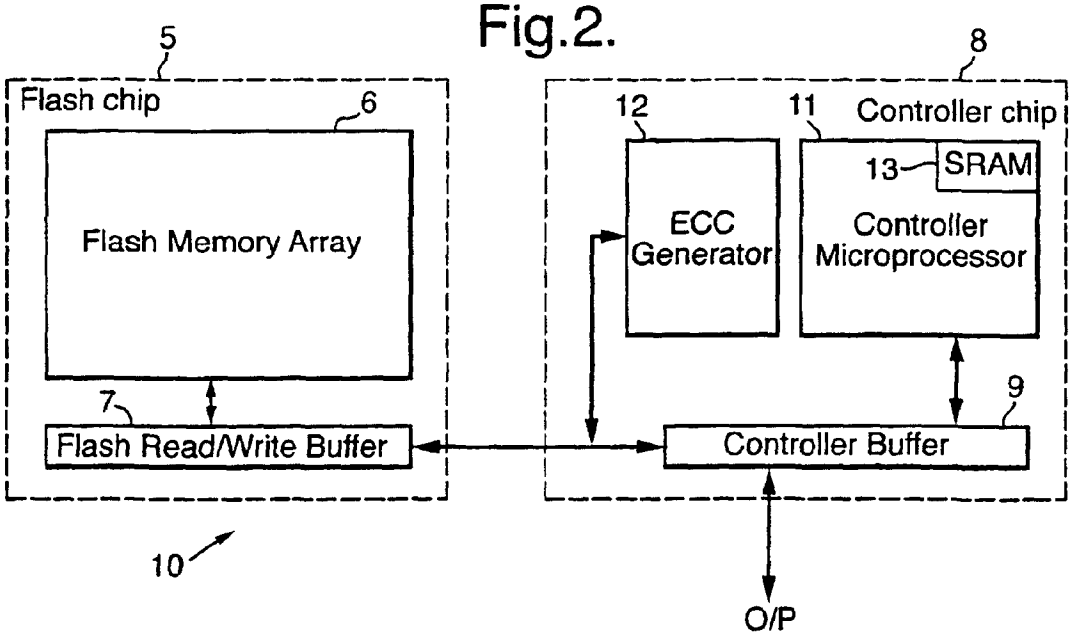


Fig.3.

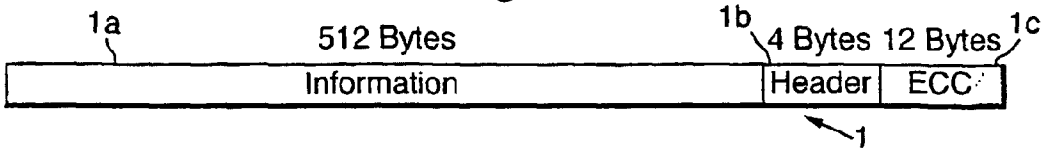


Fig.4.

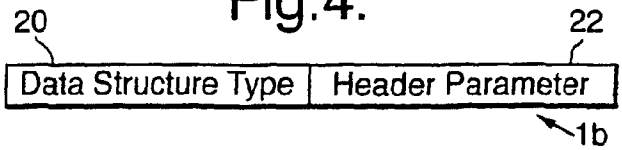


Fig.5.



Fig.6.

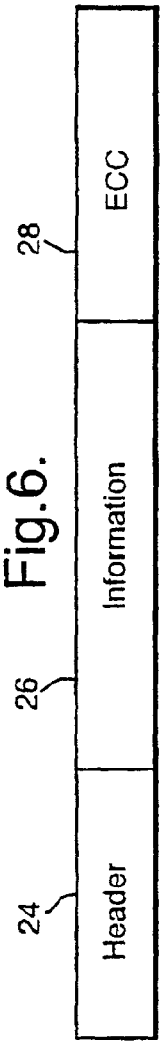


Fig.7.

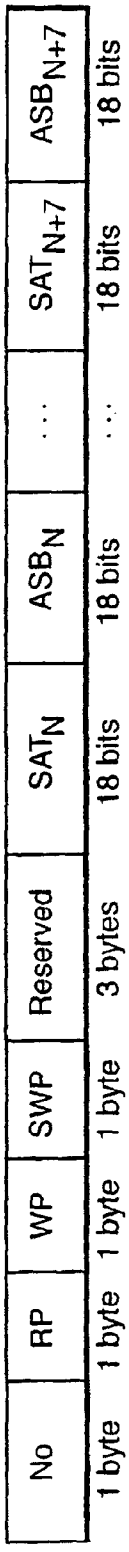


Fig.8.



12/21/2022 19:59:42
Page 38 of 57
SAMSUNG EX1002, Page 252 of 1031

Fig.9.

SAT Block Index	SAT Block Address	ASB Address	LWP	NVP	ASB Page 0	...	ASB Page N
2 bytes	3 bytes	3 bytes	1 byte	1 byte	1 byte	...	1 byte

Fig.10.

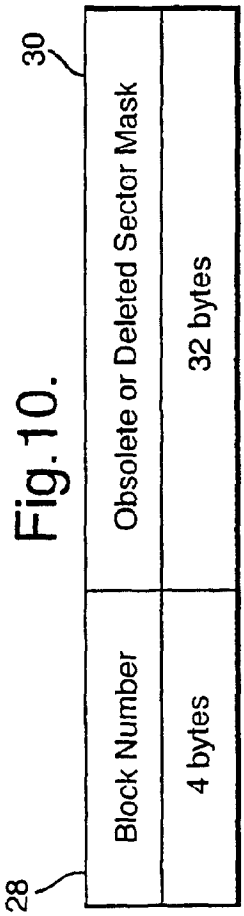
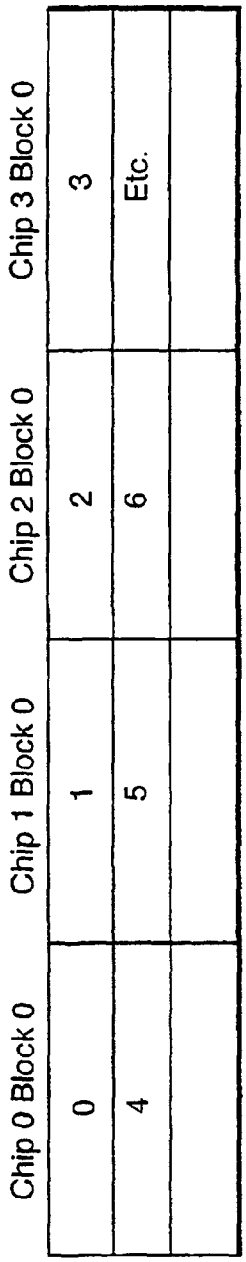


Fig.11.



12/21/2022 19:59:42
Page 39 of 57
SAMSUNG EX1002, Page 253 of 1031

Fig.12.

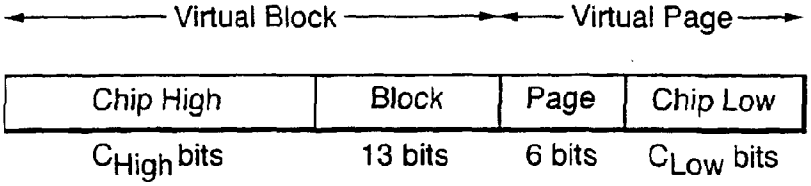


Fig.13.

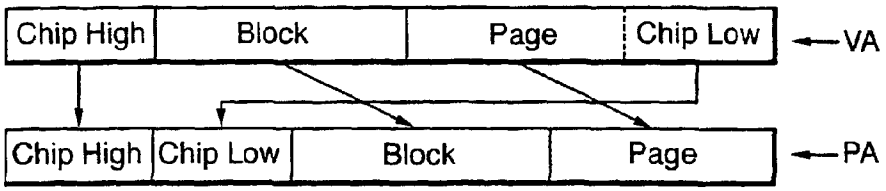
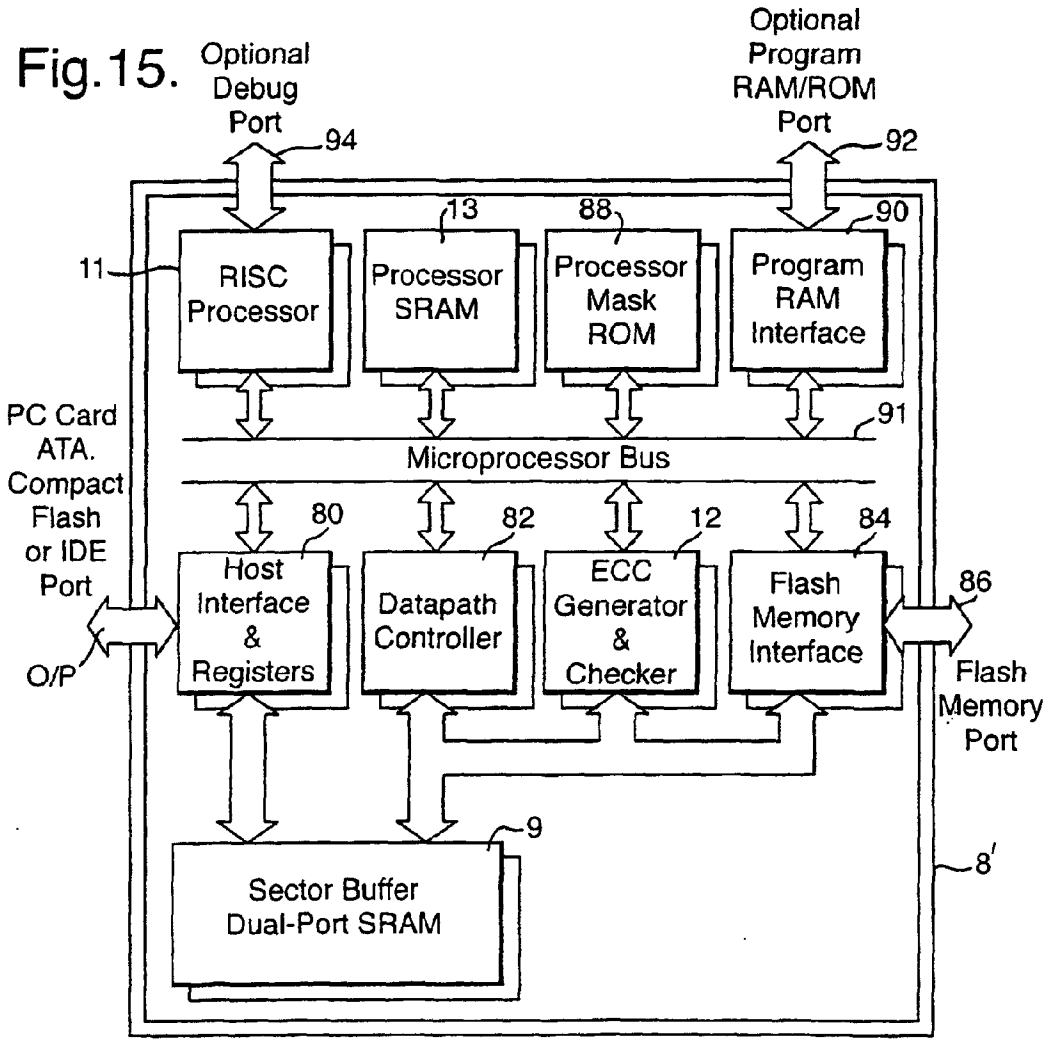


Fig.15.



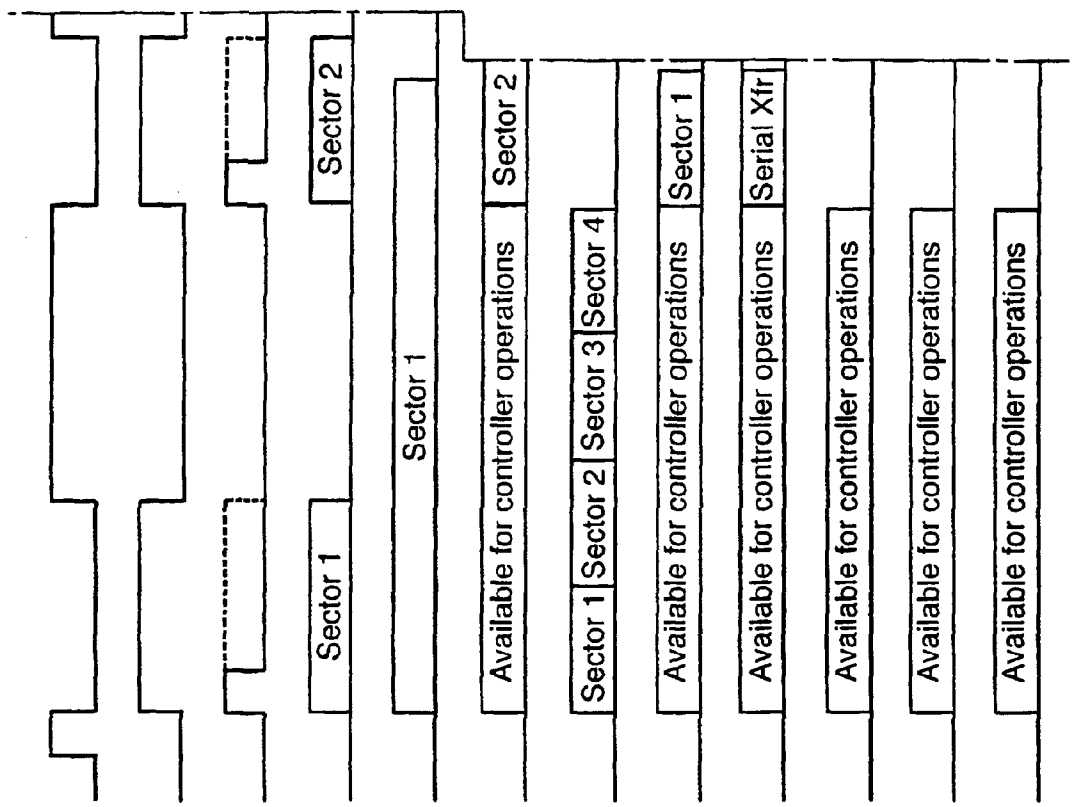
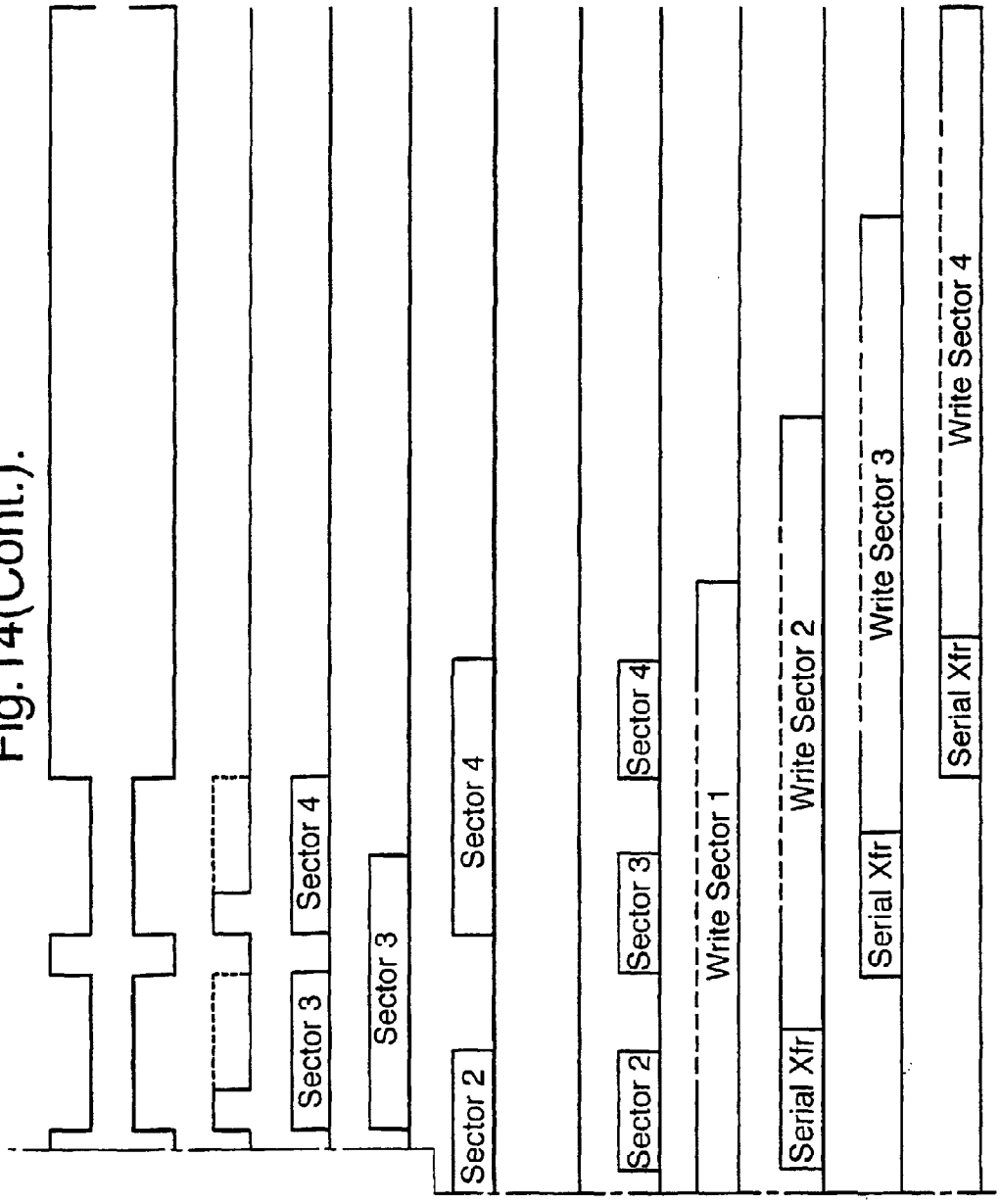


Fig.14.

SAMSUNG EX1002, Page 255 of 1031

Fig.14(Cont.).



SAMSUNG EX1002, Page 256 of 1031

Fig.16.

Capacity Allocation Table

	8MB Card		64MB Card		512MB Card	
	Number of Blocks	% of Capacity	Number of Blocks	% of Capacity	Number of Blocks	% of Capacity
Total Capacity	1024	100%	8192	100%	65536	100%
Sectors	1007	98.1%	8132	99.2%	65131	99.4%
Boot Block	1	0.1%	1	0.01%	1	0.00%
Control Block	3	0.3%	4	0.05%	4	0.00%
Sector Address Table	6	0.6%	48	0.6%	384	0.6%
Additional SAT Blocks	6	0.6%	8	0.1%	8	0.01%
Obsolete Sectors	1	0.1%	1	0.01%	1	0.00%
Erased Buffer	2	0.2%	2	0.02%	2	0.00%
Spare Blocks						

SAMSUNG EX1002, Page 257 of 1031

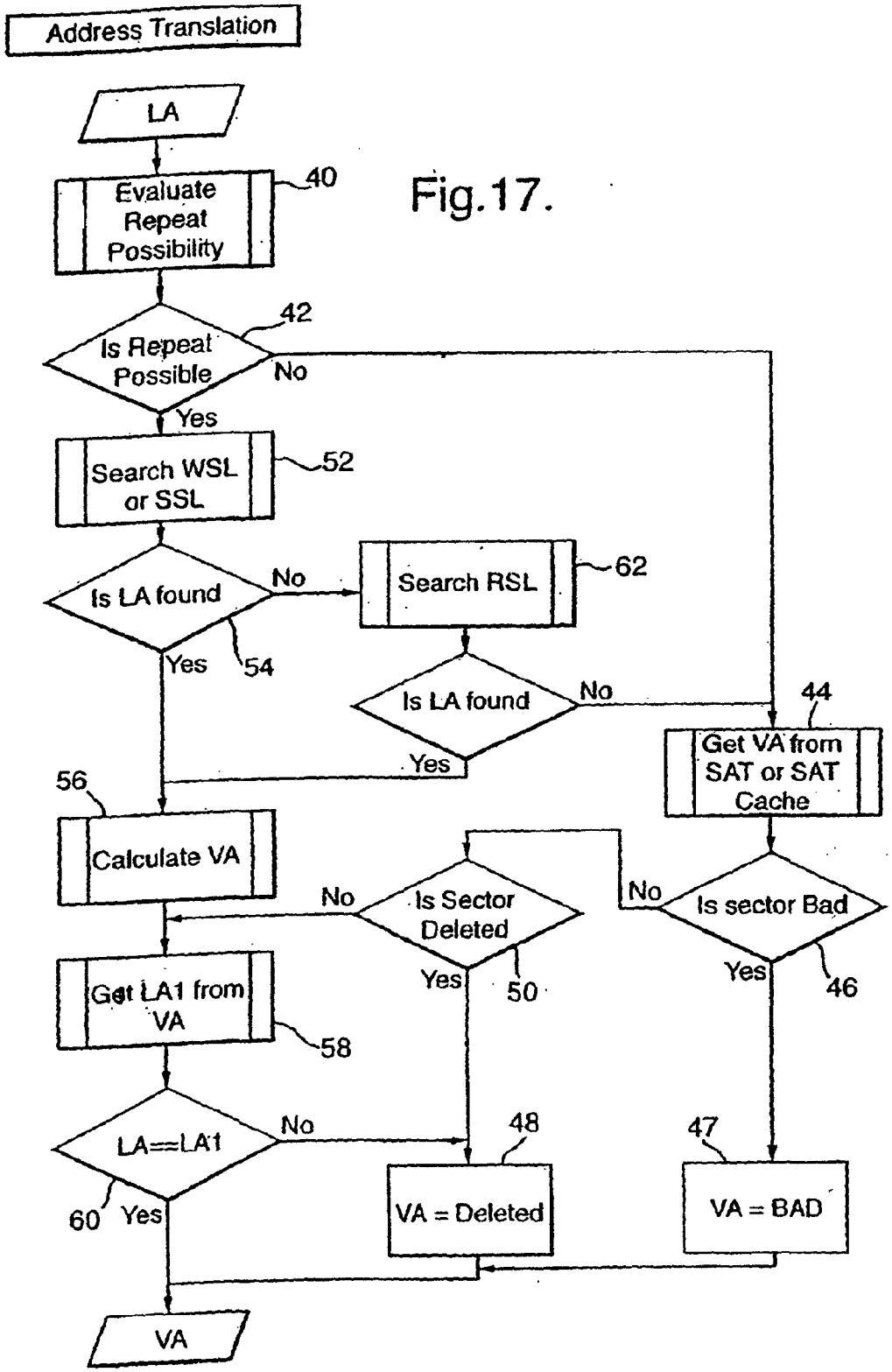


Fig.17.

SAMSUNG EX1002, Page 258 of 1031

Fig.18.

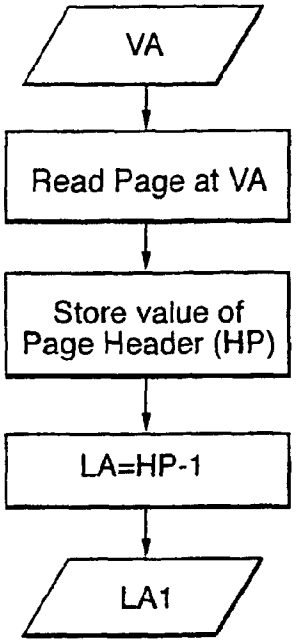
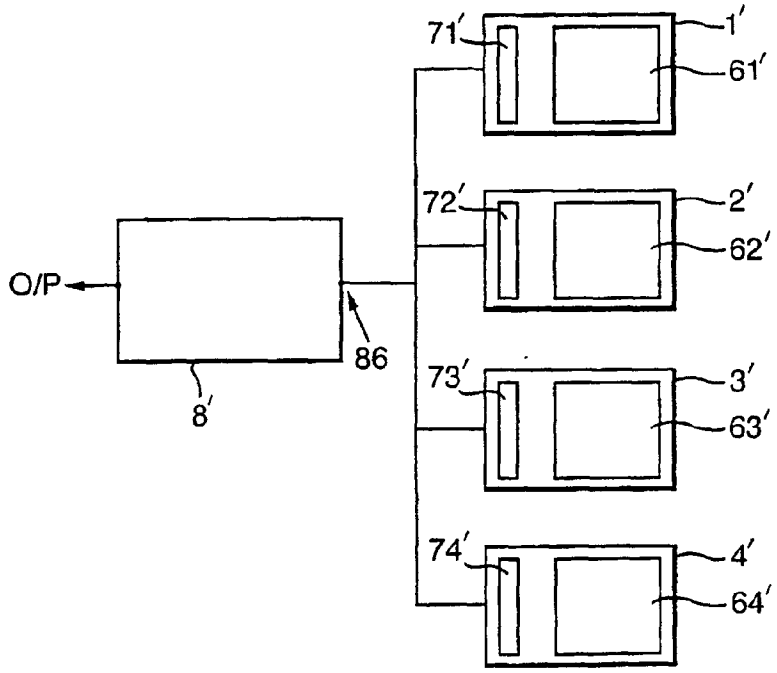


Fig.19.



12/21/2022 19:59:42
Page 45 of 57
SAMSUNG EX1002, Page 259 of 1031

Fig.20.

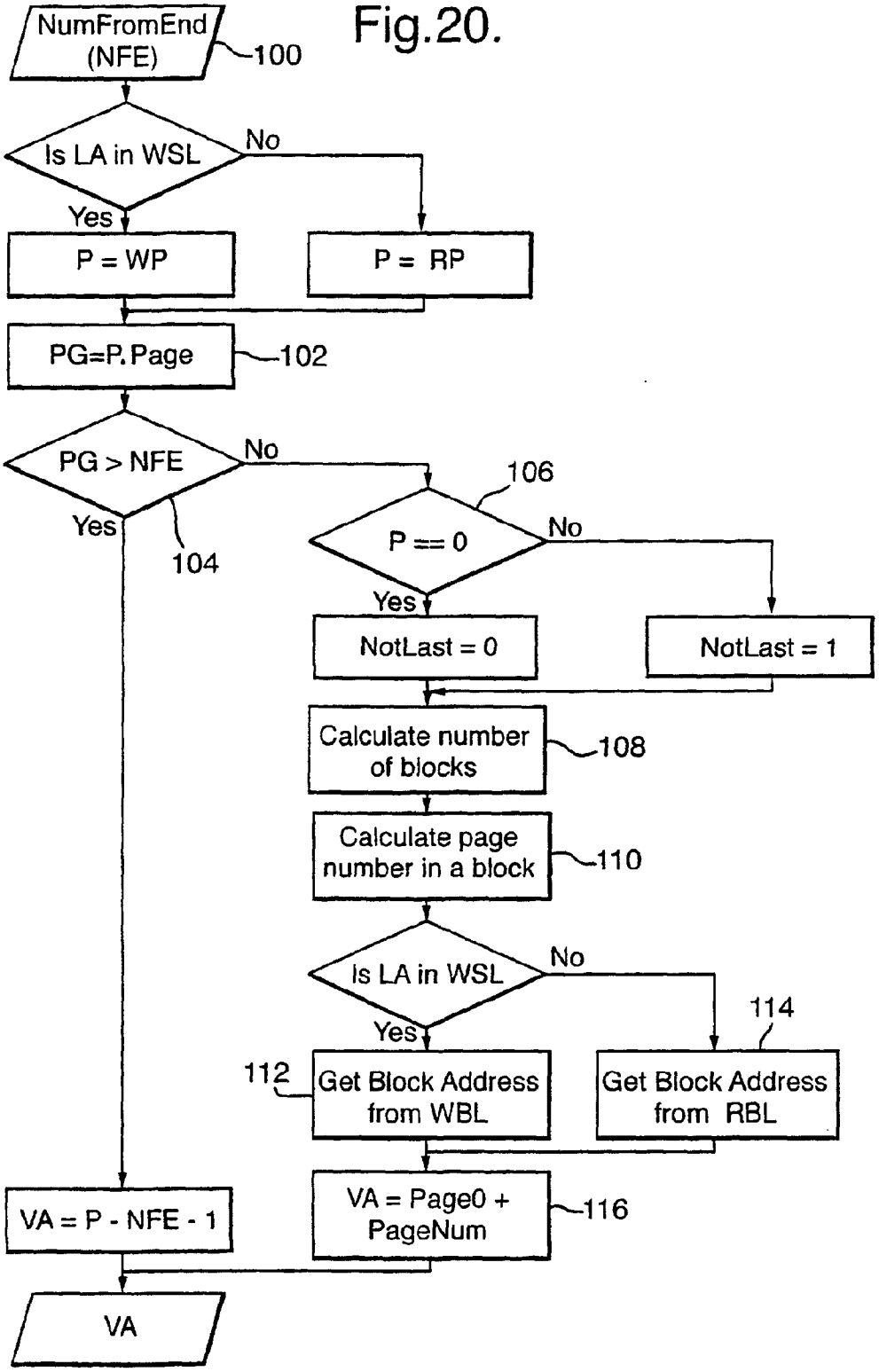

NumFromEnd (NFE)
100
Is LA in WSL
No
Yes
P = WP
P = RP
PG=P.Page
102
PG > NFE
No
Yes
104
106
P == 0
No
Yes
NotLast = 0
NotLast = 1
Calculate number of blocks
108
Calculate page number in a block
110
Is LA in WSL
No
Yes
112
Get Block Address from WBL
114
Get Block Address from RBL
VA = P - NFE - 1
VA = Page0 + PageNum
116
VA

Fig.21.

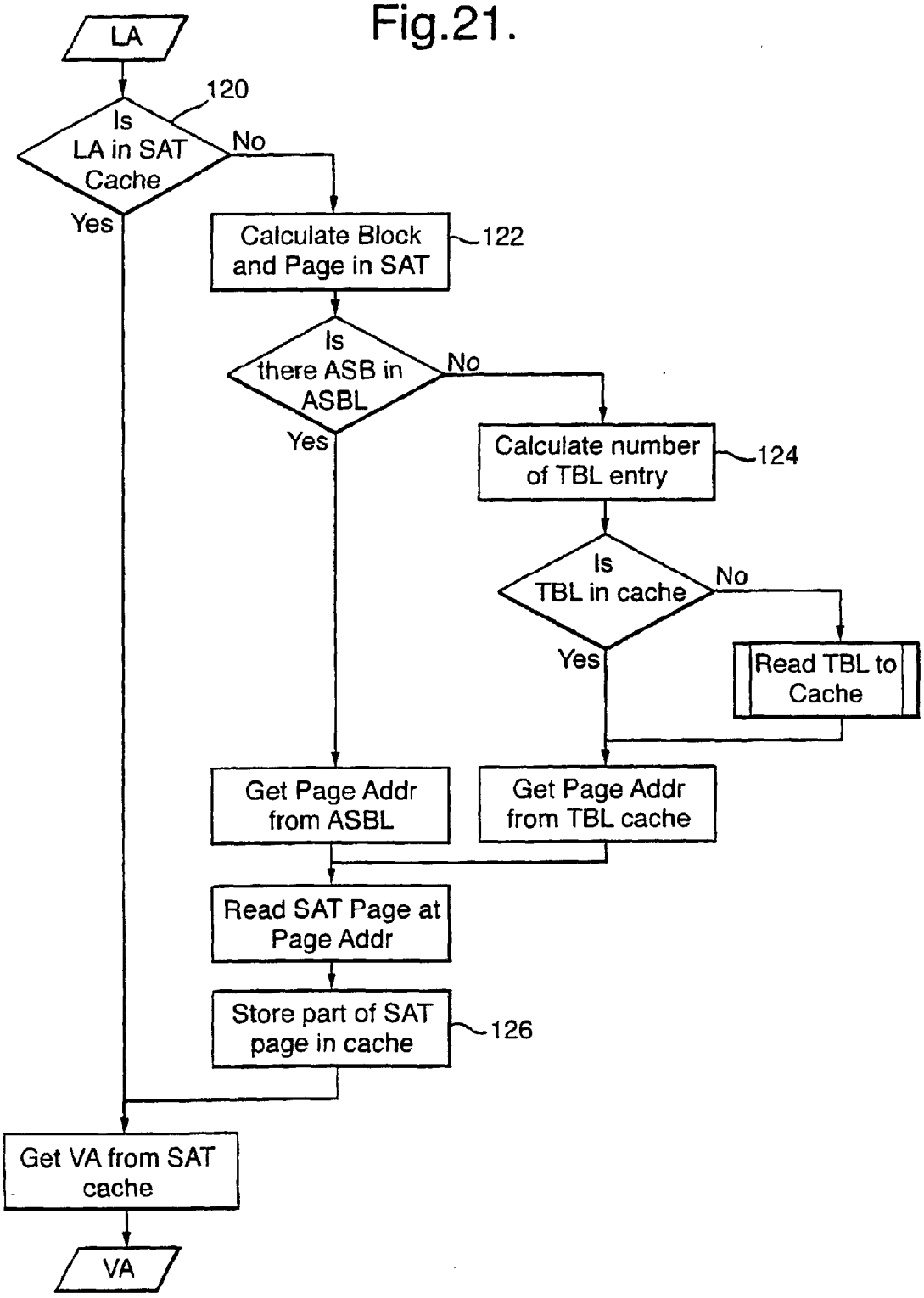
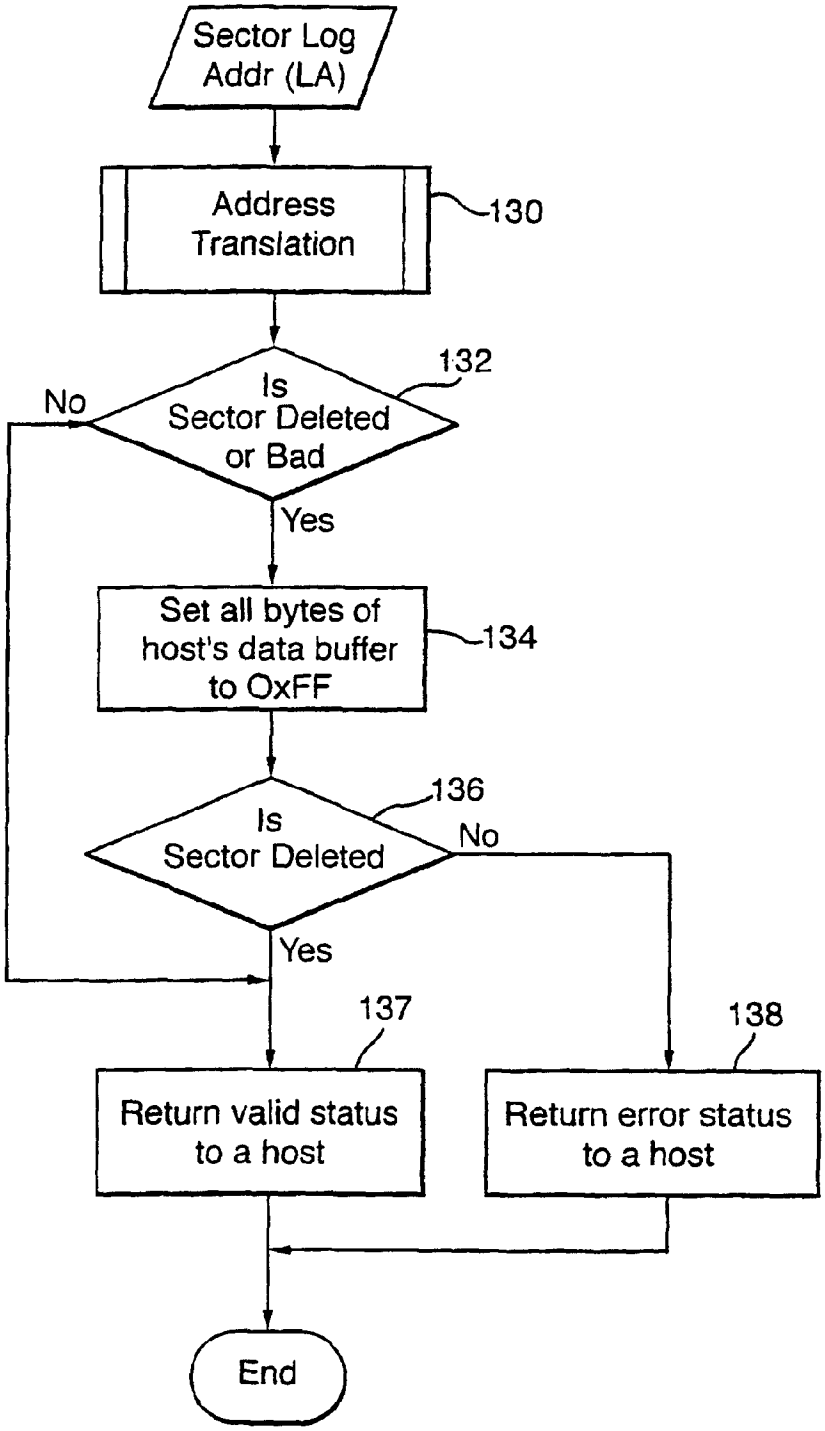
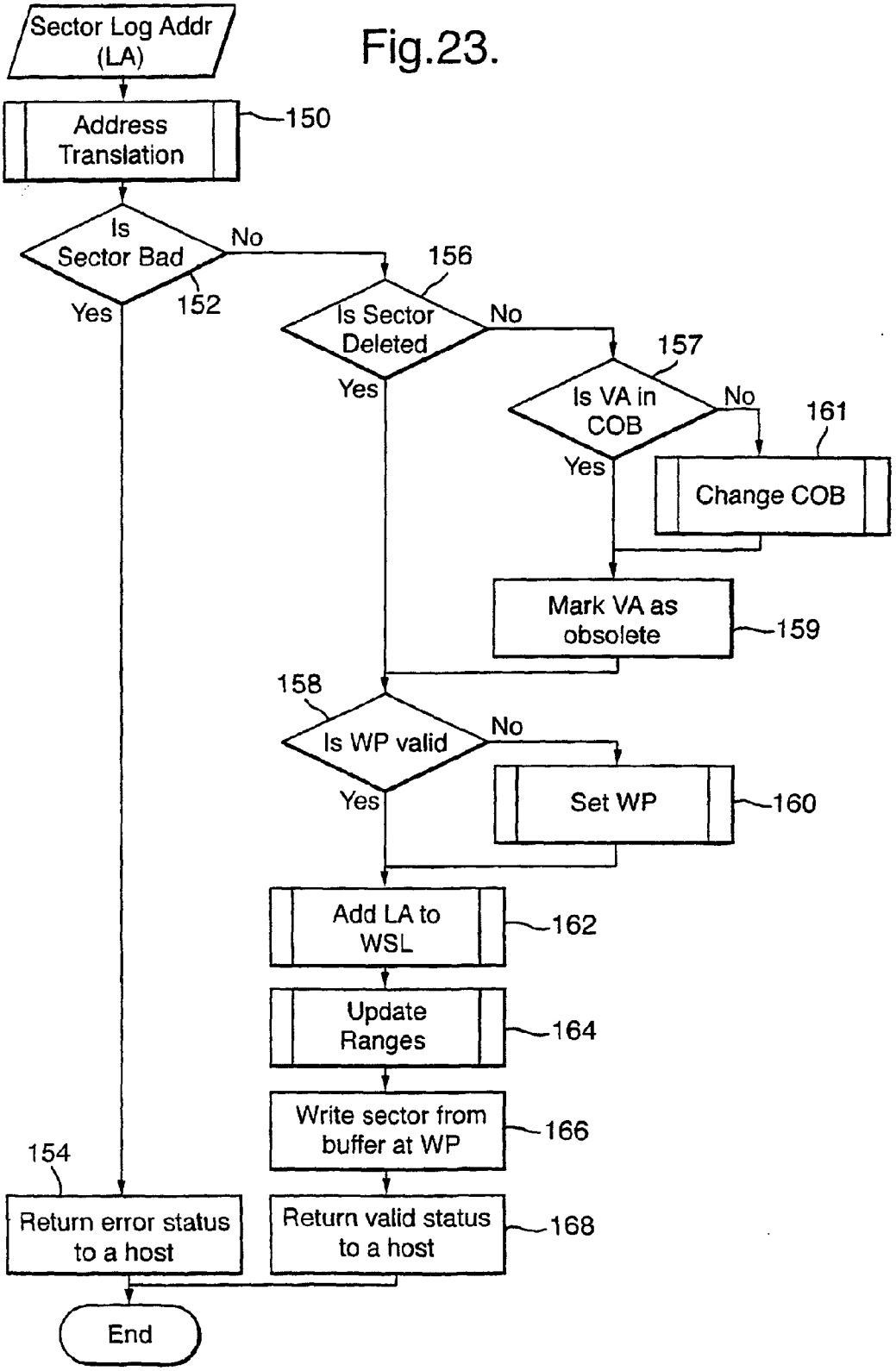

LA
120
Is LA in SAT Cache
No
Yes
Calculate Block and Page in SAT
122
Is there ASB in ASBL
No
Yes
Calculate number of TBL entry
124
Is TBL in cache
No
Yes
Read TBL to Cache
Get Page Addr from ASBL
Get Page Addr from TBL cache
Read SAT Page at Page Addr
Store part of SAT page in cache
126
Get VA from SAT cache
VA

Fig.22.



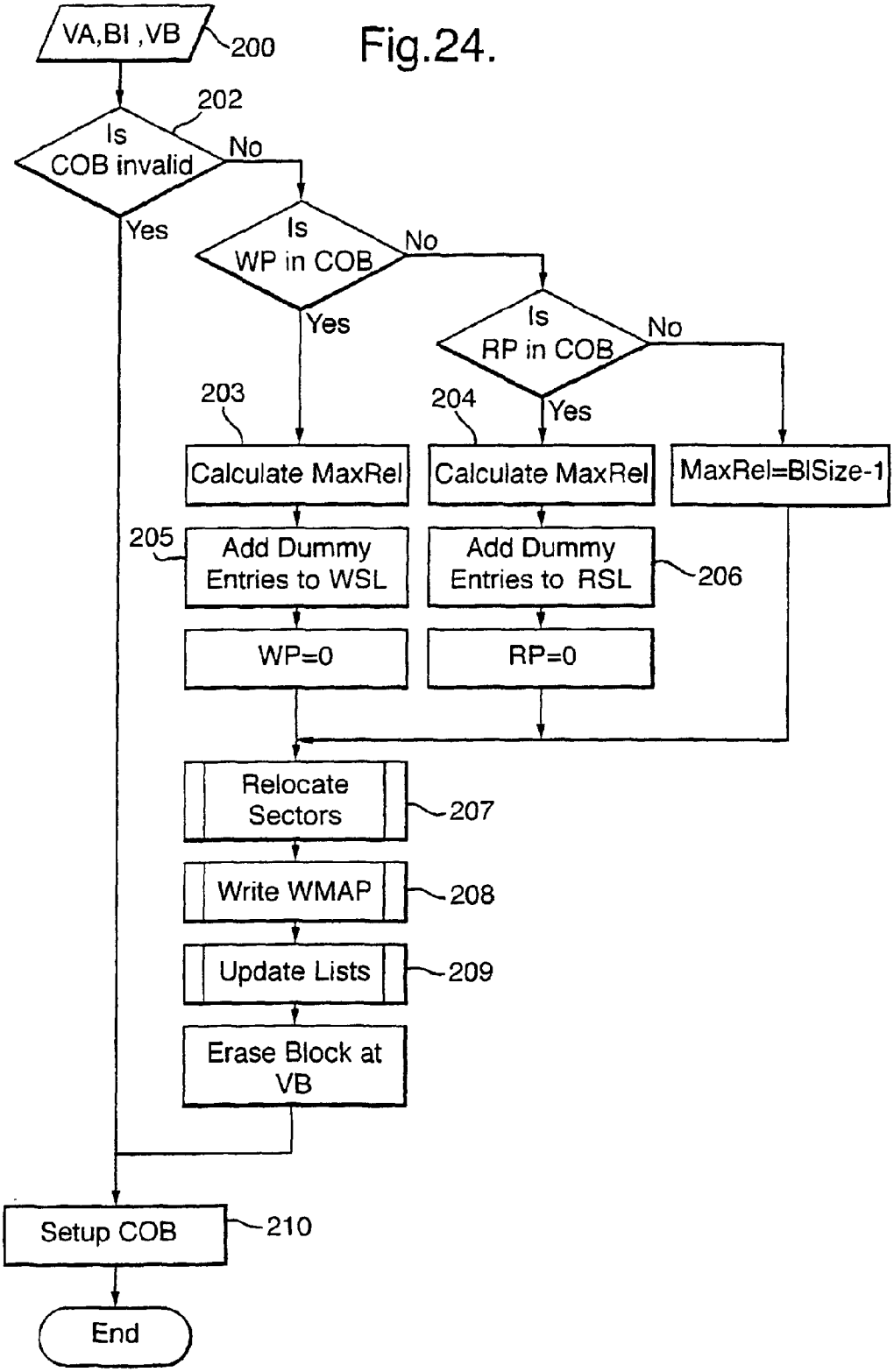
SAMSUNG EX1002, Page 262 of 1031

Fig.23.



SAMSUNG EX1002, Page 263 of 1031

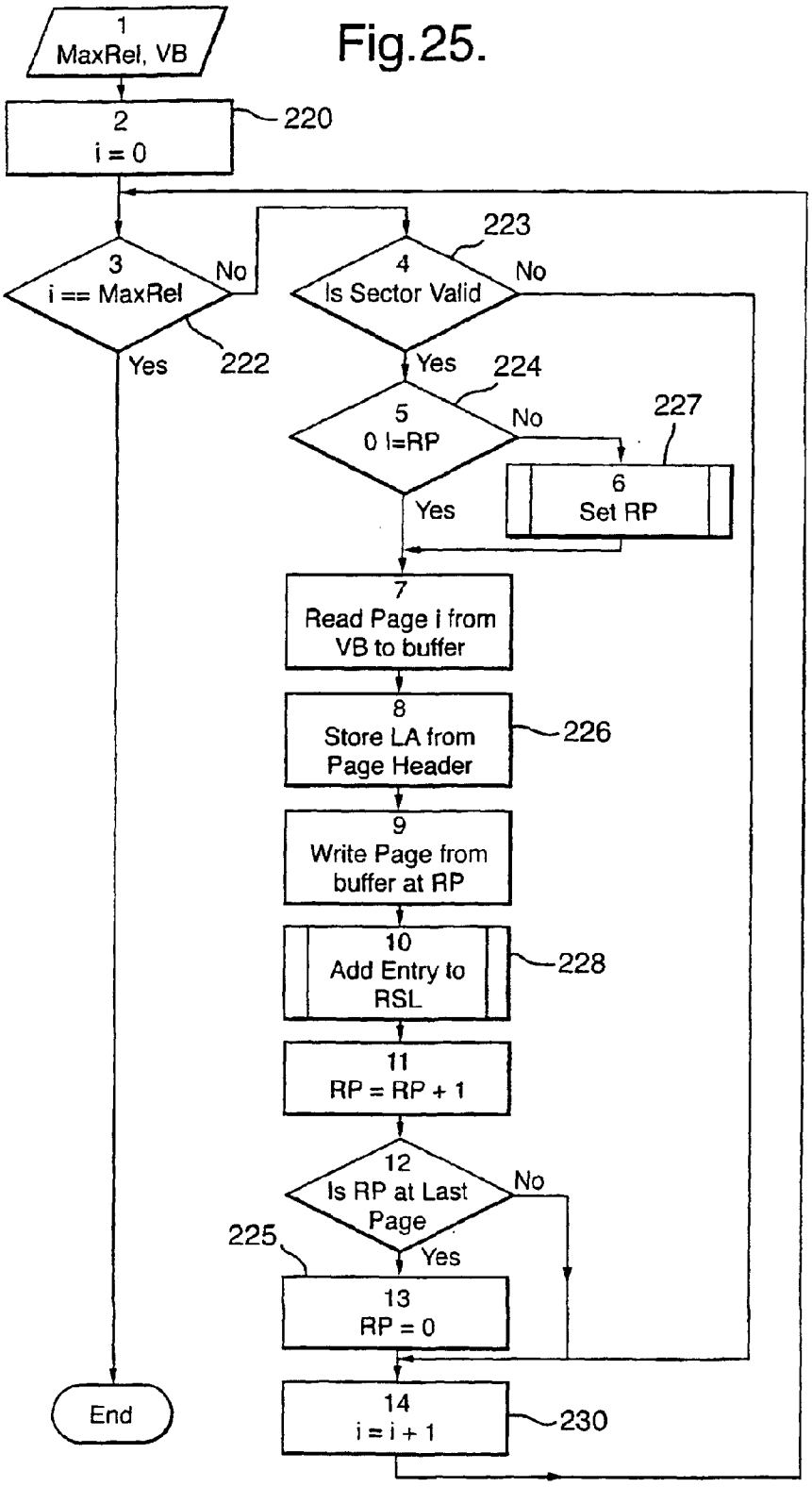
Fig.24.



12/21/2022 19:59:42

SAMSUNG EX1002, Page 264 of 1031

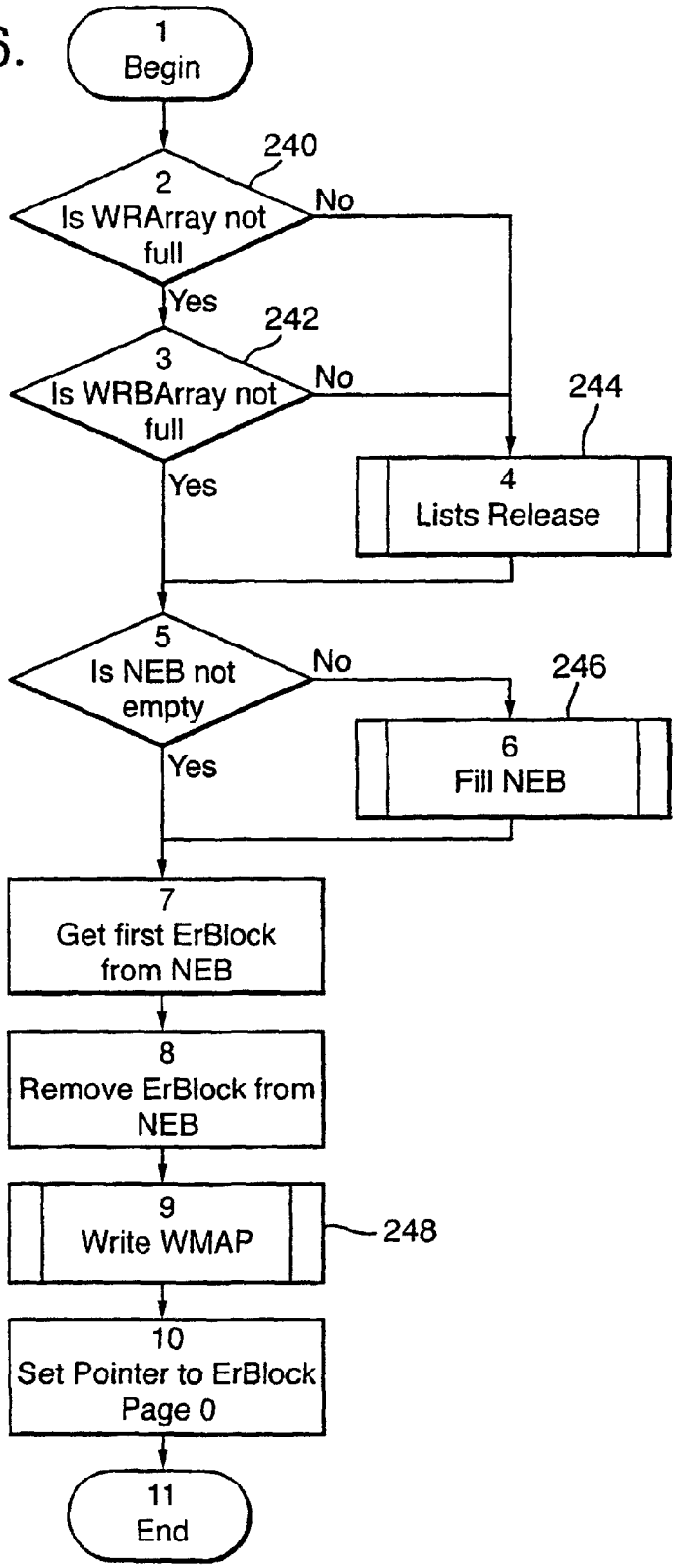
Fig.25.



12/21/2022 19:59:42

SAMSUNG EX1002, Page 265 of 1031

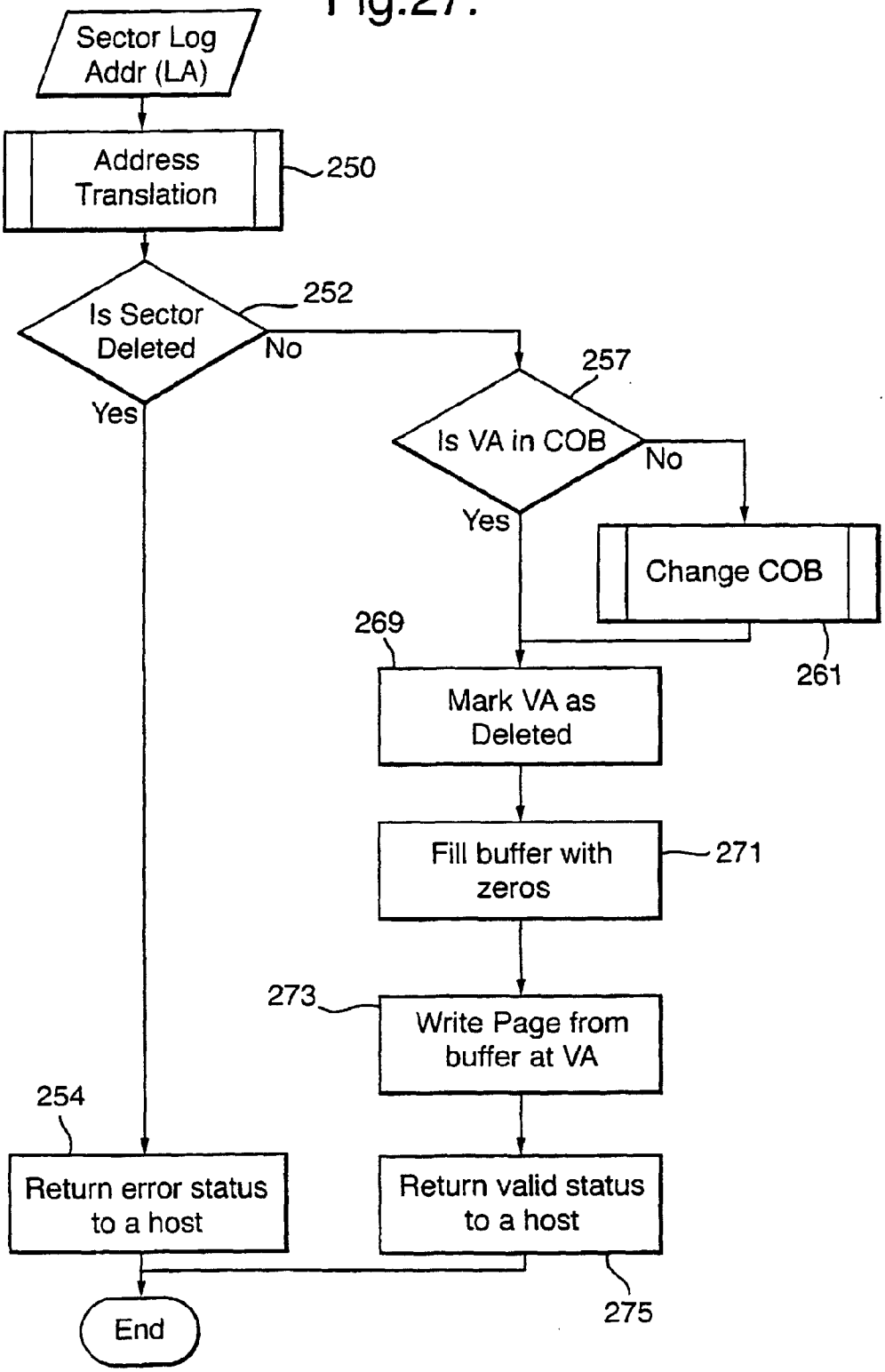
Fig.26.



12/21/2022 19:59:42

SAMSUNG EX1002, Page 266 of 1031

Fig.27.



SAMSUNG EX1002, Page 267 of 1031

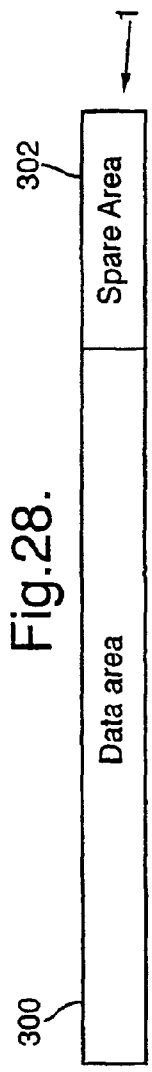


Fig.28.

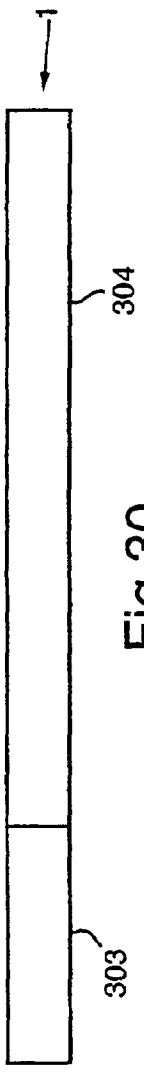


Fig.29.

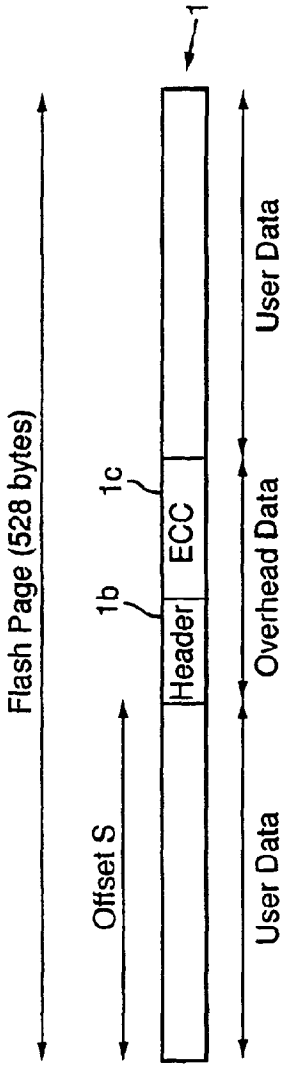


Fig.30.

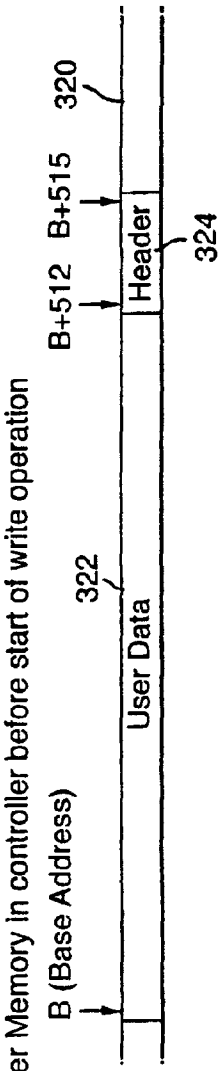


Fig.31(a).

12/21/2022 19:59:42
Page 54 of 57
SAMSUNG EX1002, Page 268 of 1031

Fig.31(b).

Data in Flash Page after completion of write operation

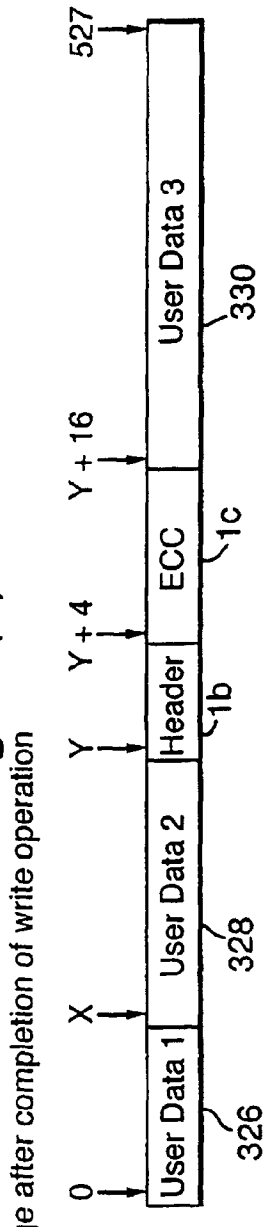


Fig.33.

Data in Buffer Memory in controller after completion of read operation

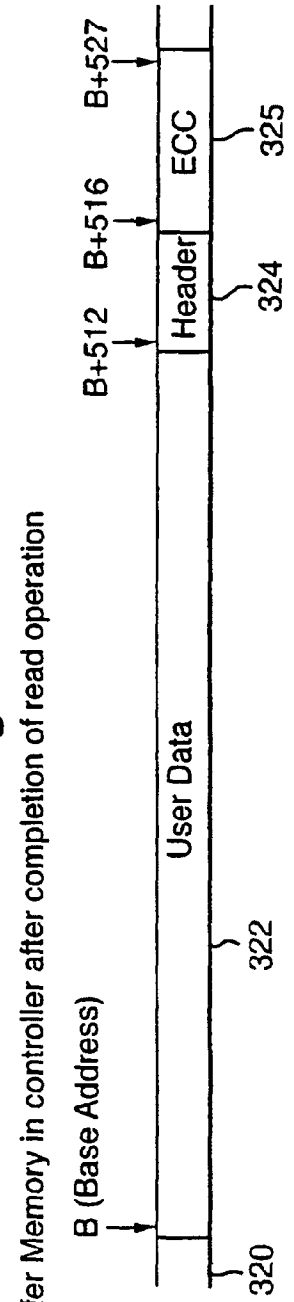
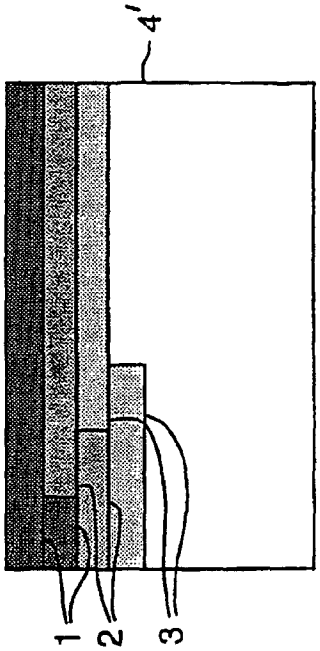


Fig.35.



12/21/2022 19:59:42

SAMSUNG EX1002, Page 269 of 1031

Fig.32.

Seqence of commands to controller data transfer hardware

Command Sequence	Command Parameters			Command Description		
	Flash Start Address	Buffer Start Address	No. of Data Xfr Cycles	ECC Generator Mode	Flash Clock	Summary
Cmnd 1	n/a	B	516	Generator on Output register disabled	off	Generate ECC for 512 bytes of User Data + Header
Cmnd 2	0	B	Y	Generator off Output register disabled	on	Transfer User Data 1 + User Data 2 to Flash buffer
Cmnd 3	continue	B + 512	4	Generator off Output register disabled	on	Transfer Header to Flash buffer
Cmnd 4	continue	n/a	12	Generator off Output register enabled	on	Transfer ECC to Flash buffer
Cmnd 5	continue	B + Y	512 - Y	Generator off Output register disabled	on	Transfer User Data 3 to Flash buffer Write Flash buffer to Flash array

SAMSUNG EX1002, Page 270 of 1031

Fig.34.

Seqence of commands to conrtoller data transfer hardware

Command Sequence	Command Parameters			Command Description		
	Flash Start Address	Buffer Start Address	No. of Data Xfr Cycles	ECC Generator Mode	Flash Clock	Summary
Cmnd 1	0	B	X	Generator on Output register disabled	on	Read Flash page to Flash buffer Transfer User Data 1 from Flash buffer
Cmnd 2	continue	B + X	Y - X	Generator on Output register disabled	on	Transfer User Data 2 from Flash buffer
Cmnd 3	continue	B + 512	16	Generator off Output register disabled	on	Transfer Header + ECC from Flash buffer
Cmnd 4	continue	B + Y	512 - Y	Generator on Output register disabled	on	Tranfer User Data 3 to Flash buffer
Cmnd 5	n/a	B + 512	16	Generator on Output register disabled	off	Transfer Header + ECC from Buffer to ECC Generator

SAMSUNG EX1002, Page 27 of 1031

1. IEEE Standard for Information Technology - POSIX(R) Ada Language Interfaces - Part 1: Binding for System Application Program Interface (API) - Amendment 2: Protocol-Independent Interfaces

IEEE XploreIEEE Standards1999-01-014IEEE Std 1003.5, 1999 Edition (Page(s): 0_1)

2. IEEE Standard for Information Technology--POSIX(TM) Ada Language Interfaces--Part 1: Binding for System Application Program Interface (API)--Amendment 1: Realtime Extensions

IEEE XploreIEEE Standards1997-11-303IEEE Std 1003.5b-1996 (Includes IEEE Std 1003.5-1992) (Page(s): 1-550)

3. SAS 9.1.3 Language Reference Concepts

IPCOM000243217DLegacy Journals2004-12-313SAS Institute Inc.

4. SAS 9.1 Language Reference Concepts

IPCOM000243169DLegacy Journals2004-12-313SAS Institute Inc.

5. An overview of logic architectures inside flash memory devices

IEEE XploreIEEE Periodicals2003-04-013Proceedings of the IEEE

6. Hydra: A Block-Mapped Parallel Flash Memory Solid-State Disk Architecture

IEEE XploreIEEE Periodicals2010-06-253IEEE Transactions on Computers

7. IEEE Standard for Scalable Storage Interface

IEEE XploreIEEE Standards2006-03-223IEEE Std 1285-2005 (Page(s): 1-140)

8. Ozone (O3): An Out-of-Order Flash Memory Controller Architecture

IEEE XploreIEEE Periodicals2010-10-073IEEE Transactions on Computers

9. Detecting Solid-State Disk Geometry for Write Pattern Optimization

IEEE XploreIEEE Conferences2011-08-0132011 IEEE 17th International Conference on Embedded and Real-Time Computing Systems and Applications

10. Conference Record of the Twenty First IEEE Photovoltaic Specialists Conference - 1990 (Cat. No.90CH2838-1)

IEEE XploreIEEE Conferences1990-01-013IEEE Conference on Photovoltaic Specialists (Page(s): 0_1)

11. RAP.2---An Associative Processor for Databases and Its Applications

IEEE XploreIEEE Periodicals1979-06-013IEEE Transactions on Computers

12. An Application of a Large-Image Memory Graphics Workstation

IEEE XploreIEEE Periodicals1985-02-013IEEE Computer Graphics and Applications

13. Formalising Flash Memory: First Steps

IEEE XploreIEEE Conferences2007-07-01312th IEEE International Conference on Engineering Complex Computer Systems (ICECCS 2007) (Page(s): 251-260)

14. Advanced Nonvolatile Memory Designs and Technologies

IEEE XploreWiley-IEEE eBooks2002-01-013Wiley-IEEE Press 2002

15. A virtual machine time-sharing system

IEEE XploreIBM Periodicals1970-01-013IBM Systems Journal

16. SAS 9.1 Companion for OpenVMS Alpha

IPCOM000243141DLegacy Journals2004-12-313SAS Institute Inc.

17. SAS® 9.1 Companion for OpenVMS Alpha

IPCOM000243272DLegacy Journals2004-12-313SAS Institute Inc.

18. Performance comparison of logical-address-to-physical-address algorithms for non-volatile memory

IEEE XploreIEEE Conferences2003-01-013ASIC, 2003. Proceedings. 5th International Conference on

19. SAS® 9.1 Companion for UNIX Environments

IPCOM000243153DLegacy Journals2004-12-313SAS Institute Inc.

20. SAS 9.1 Companion for UNIX Environments

IPCOM000243336DLegacy Journals2004-12-313SAS Institute Inc.

21. Format for a graphical communication protocol (RFC0965)

IPCOM000004961DInternet Society RFC1985-12-013Internet Society Requests for Comments

22. IEEE Recommended Practice for Futurebus+(R)

IEEE XploreIEEE Standards1994-07-253IEEE Std 896.3-1993 (Page(s): 1-252)

23. Microprocessors in brief

IEEE XploreIBM Periodicals1985-03-013IBM Journal of Research and Development

24. IBM System Storage DS5000 Series Hardware Guide

SG24802300IBM Redbooks2013-01-103

25. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries

IEEE XploreIEEE Standards1991-01-183IEEE Std 610 (Page(s): 1-217)

SAMSUNG EX1002, Page 273 of 1031

26. SAS 9.3 Companion for Windows

IPCOM000243174DLegacy Journals2011-12-31SAS Institute Inc.

27. SAS®9 Companion for OpenVMS Alpha

IPCOM000244051DLegacy Journals2002-12-31SAS Institute Inc.

28. A space-efficient flash translation layer for CompactFlash systems

IEEE XploreIEEE Periodicals2002-05-01IEEE Transactions on Consumer Electronics

29. Whats New in SAS 9.2

IPCOM000243172DLegacy Journals2009-12-31SAS Institute Inc.

30. Office-by-Example: A business language that unifies data and word processing and electronic mail

IEEE XploreIBM Periodicals1982-01-01IBM Systems Journal

31. SAS®9 Companion for Microsoft Windows

IPCOM000244068DLegacy Journals2002-12-31SAS Institute Inc.

32. Optimized structure for fast data storage to non-volatile memory with minimum chip size

IEEE XploreIEEE Conferences2004-01-01Proceedings 7th International Conference on Signal Processing, 2004. Proceedings. ICSP '04. 2004.

33. IEEE Guide to the POSIX Open System Environment (OSE)

IEEE XploreIEEE Standards1995-01-01IEEE Std 1003.0-1995 (Page(s): 0_3)

34. SAS 9.1 Companion for Windows

IPCOM000243180DLegacy Journals2004-12-31SAS Institute Inc.

35. SAS® 9.1 Companion for Windows

IPCOM000244072DLegacy Journals2004-12-31SAS Institute Inc.

36. Requirements for Internet Hosts - Application and Support (RFC1123)

IPCOM000001933DInternet Society RFC1989-10-01Internet Society Requests for Comments

SAMSUNG EX1002, Page 274 of 1031

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
16 June 2005 (16.06.2005)

PCT

(10) International Publication Number
WO 2005/055043 A1

(51) International Patent Classification[7]: G06F 3/06

(21) International Application Number:
PCT/US2004/039306

(22) International Filing Date:
22 November 2004 (22.11.2004)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10/722,614 26 November 2003 (26.11.2003) US

(71) Applicant *(for all designated States except US)*: **VERITAS OPERATING CORPORATION** [US/US]; 350 Ellis Street, Mountain View, CA 94043 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **KARR, Ronald, S.** [US/US]; 333 Ramona Street, Palo Alto, CA 94301 (US). **KISELEV, Oleg** [US/US]; 555 Bryant #333, Palo Alto, CA 94301 (US). **MIROSCHNICHENKO, Alex** [US/US]; 239 Brannon #11E, San Francisco, CA 94107 (US). **KONG, Algaia** [CA/US]; 2192 Foxtail Court, San Jose, CA 95138 (US).

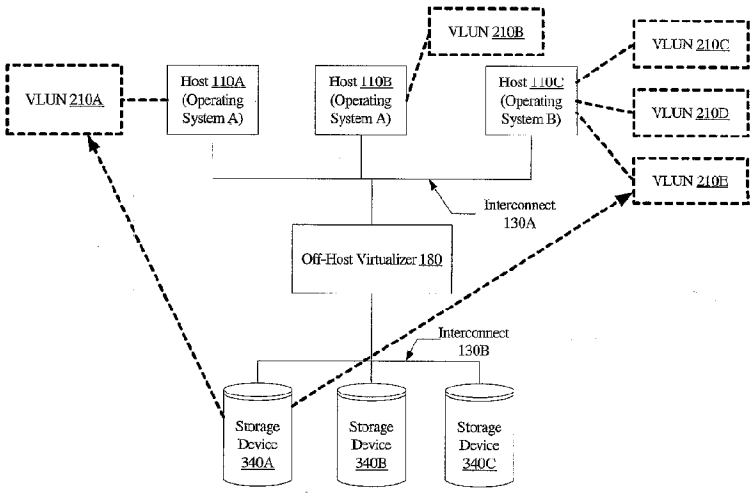
(74) Agent: **KIVLIN, B., Noel**; MEYERTONS, HOOD, KIVLIN, KOWERT & GOETZEL, P.C., P.O. Box 398, Austin, TX 78767-0398 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, 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, NA, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: SYSTEM AND METHOD FOR EMULATING OPERATING SYSTEM METADATA TO PROVIDE CROSS-PLATFORM ACCESS TO STORAGE VOLUMES



(57) Abstract: A system and method for emulating operating system metadata to provide cross-platform access to storage volumes may include a first host and an off-host virtualizer. The off-host virtualizer may be configured to generate operating system metadata for a first virtual storage device, such as a virtual LUN, and to make the operating system metadata accessible to the first host. The metadata may be customized in accordance with a requirement of the operating system in use at the first host. The first host may include a storage software stack including a first layer, configured to use the operating system metadata to detect the existence of the first virtual storage device as an addressable storage device.

WO 2005/055043 A1

SAMSUNG EX1002, Page 275 of 1031

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

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.

TITLE: SYSTEM AND METHOD FOR EMULATING OPERATING SYSTEM METADATA TO PROVIDE CROSS-PLATFORM ACCESS TO STORAGE VOLUMES

BACKGROUND OF THE INVENTION

Field of the Invention

[0001] This invention relates to computer systems and, more particularly, to off-host virtualization within storage environments.

Description of the Related Art

[0002] Many business organizations and governmental entities rely upon applications that access large amounts of data, often exceeding a terabyte of data, for mission-critical applications. Often such data is stored on many different storage devices, which may be heterogeneous in nature, including many different types of devices from many different manufacturers.

[0003] Configuring individual applications that consume data, or application server systems that host such applications, to recognize and directly interact with each different storage device that may possibly be encountered in a heterogeneous storage environment would be increasingly difficult as the environment scaled in size and complexity. Therefore, in some storage environments, specialized storage management software and hardware may be used to provide a more uniform storage model to storage consumers. Such software and hardware may also be configured to present physical storage devices as virtual storage devices (e.g., virtual SCSI disks) to computer hosts, and to add storage features not present in individual storage devices to the storage model. For example, features to increase fault tolerance, such as data mirroring, snapshot/fixed image creation, or data parity, as well as features to increase data access performance, such as disk striping, may be implemented in the storage model via hardware or software. The added storage features may be referred to as storage virtualization features, and the software and/or hardware providing the virtual storage devices and the added storage features may be termed "virtualizers" or "virtualization controllers". Virtualization may be performed within computer hosts, such as within a volume manager layer of a storage software stack at the host, and/or in devices external to the host, such as virtualization switches or virtualization appliances. Such external devices providing virtualization may be termed "off-host" virtualizers, and may be utilized in order to offload processing required for virtualization from the host. Off-host virtualizers may be connected to the external physical storage devices for which they provide virtualization functions via a variety of interconnects, such as Fiber Channel links, Internet Protocol (IP) networks, and the like.

[0004] Traditionally, storage software within a computer host consists of a number of layers, such as a file system layer, a disk driver layer, etc. Some of the storage software layers may form part of the operating system in use at the host, and may differ from one operating system to another. When accessing a physical disk, a layer such as the disk driver layer for a given operating system may be configured to expect certain types of configuration information for the disk to be laid out in a specific format, for example in a header (located at the first few blocks of the disk) containing disk partition layout information. The storage stack software layers used to access local physical disks may also be utilized to access external storage devices presented as virtual storage devices by off-host virtualizers. Therefore, it may be desirable for an off-host virtualizer to provide configuration information for the

12/21/2022 19:09:40 Page 3 of 52
SAMSUNG EX1002, Page 277 of 1031

virtual storage devices in formats expected by the different operating systems that may be in use at the hosts in the storage environment. In addition, it may be desirable for the off-host virtualizer to implement a technique to flexibly and dynamically map storage within external physical storage devices to the virtual storage devices presented to the host storage software layers, e.g., without requiring a reboot of the host.

SUMMARY OF THE INVENTION

[0005] Various embodiments of a system and method for emulating operating system metadata to provide cross-platform access to storage volumes are disclosed. According to a first embodiment, a system may include a first host and an off-host virtualizer, such as a virtualization switch or a virtualization appliance. The off-host virtualizer may be configured to generate operating system metadata for a first virtual storage device, such as a virtual LUN, and to make the operating system metadata accessible to the first host. The operating system metadata may be customized according to a requirement of the operating system in use at the host; that is, the off-host virtualizer may be capable of providing metadata customized for a number of different operating systems in use at different hosts. The first host may be configured to specify the specific metadata requirements of its operating system to the off-host virtualizer. The first host may include a storage software stack including a first layer, such as a disk driver layer, configured to use the operating system metadata provided by the off-host virtualizer to detect the existence of the first virtual storage device as an addressable storage device such as a partition. In another embodiment, the off-host virtualizer may be configured to aggregate storage within one or more physical storage devices into a logical volume, and to map the logical volume to the first virtual storage device or virtual LUN. In such an embodiment, the off-host virtualizer may also be configured to provide logical volume metadata to a second layer of the storage software stack (such as an intermediate driver layer between a disk driver layer and a file system layer), allowing the second layer to perform I/O operations on the logical volume.

[0006] In one embodiment, the first virtual storage device may be initially unmapped to physical storage. The recognition of the unmapped first virtual storage device as an addressable storage device may occur during a system initialization stage prior to an initiation of production I/O operations. In this way, an unmapped or "blank" virtual LUN may be prepared for subsequent dynamic mapping by the off-host virtualizer. The unmapped LUN may be given an initial size equal to the maximum allowed LUN size supported by the operating system in use at the host, so that the size of the virtual LUN may not require modification after initialization. In some embodiments, multiple virtual LUNs may be pre-generated for use at a single host, for example in order to isolate storage for different applications, or to accommodate limits on maximum LUN sizes.

[0007] In another embodiment, the system may also include two or more physical storage devices, and the off-host virtualizer may be configured to dynamically map physical storage from a first and a second physical storage device to a respective range of addresses within the first virtual storage device. For example, the off-host virtualizer may be configured to perform an N-to-1 mapping between the physical storage devices (which may be called physical LUNs) and virtual LUNs, allowing storage in the physical storage devices to be accessed from the host via the pre-generated virtual LUNs. Configuration information regarding the location of the first and/or the second address ranges within the virtual LUN (i.e., the regions of the virtual LUN that are mapped to the physical storage devices) may be passed from the off-host virtualizer to a second layer of the storage stack at the host (e.g., an intermediate driver layer above a disk driver layer) using a variety of different mechanisms. Such mechanisms may

12/21/2022 19:09:40 Page 4 of 52
SAMSUNG EX1002, Page 278 of 1031

include, for example, the off-host virtualizer writing the configuration information to certain special blocks within the virtual LUN, sending messages to the host over a network, or special extended SCSI mode pages. In one embodiment, two or more different ranges of physical storage within a single physical storage device may be mapped to corresponding pre-generated virtual storage devices such as virtual LUNs and presented to corresponding hosts. That is, the off-host virtualizer may allow each host of a plurality of hosts to access a respective portion of a physical storage device through a respective virtual LUN. In such embodiments, the off-host virtualizer may also be configured to implement a security policy isolating the ranges of physical storage within the shared physical storage device; i.e., to allow a host to access only those regions to which the host has been granted access, and to prevent unauthorized accesses.

[0008] In one embodiment, the off-host virtualizer may be further configured to aggregate storage within a physical storage device into a logical volume, dynamically map the logical volume to a range of addresses within a pre-generated virtual storage device, and pass logical volume metadata to the second layer of the storage stack, allowing I/O operations to be performed on the logical volume. Storage from a single physical storage device may be aggregated into any desired number of different logical volumes, and any desired number of logical volumes may be mapped to a single virtual storage device or virtual LUN. The off-host virtualizer may be further configured to provide volume-level security, i.e., to prevent unauthorized access from a host to a logical volume, even when the physical storage corresponding to the logical volume is part of a shared physical storage device. In addition, physical storage from any desired number of physical storage devices may be aggregated into a logical volume, thereby allowing a single volume to extend over a larger address range than the maximum allowed size of a single physical storage device. The virtual storage devices or virtual LUNs may be distributed among a number of independent front-end storage networks, such as fiber channel fabrics, and the physical storage devices backing the logical volumes may be distributed among a number of independent back-end storage networks. For example, a first host may access its virtual storage devices through a first storage network, and a second host may access its virtual storage devices through a second storage network independent from the first (that is, reconfigurations and/or failures in the first storage network may not affect the second storage network). Similarly, the off-host virtualizer may access a first physical storage device through a third storage network, and a second physical storage device through a fourth storage network. The ability of the off-host virtualizer to dynamically map storage across pre-generated virtual storage devices distributed among independent storage networks may support a robust and flexible storage environment.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] FIG. 1a is a block diagram illustrating one embodiment of a computer system.

[0010] FIG. 1b is a block diagram illustrating an embodiment of a system configured to utilize off-host block virtualization.

[0011] FIG. 2a is a block diagram illustrating the addition of operating-system specific metadata to a virtual logical unit (LUN) encapsulating a source volume, according to one embodiment.

[0012] FIG. 2b is a block diagram illustrating an embodiment where an off-host virtualizer is configured to include a partition table within operating system metadata.

SAMSUNG EX1002, Page 279 of 1031

[0013] FIG. 2c is a block diagram illustrating an embodiment where an off-host virtualizer is configured to aggregate storage within physical storage devices into a logical volume and to map the logical volume to a virtual LUN.

[0014] FIG. 3 is a block diagram illustrating an embodiment including an off-host virtualizer configured to create a plurality of virtual LUNs.

[0015] FIG. 4 is a flow diagram illustrating aspects of the operation of a host and an off-host virtualizer configured to generate operating system metadata.

[0016] FIG. 5 is a flow diagram illustrating an embodiment where an off-host virtualizer is configured to aggregate storage in one or more physical storage devices into a logical volume, and to map the logical volume to a virtual LUN.

[0017] FIG. 6 is a block diagram illustrating an example of an unmapped virtual LUN according to one embodiment.

[0018] FIG. 7 is a block diagram illustrating an embodiment including an off-host virtualizer configured to create a plurality of unmapped virtual LUNs.

[0019] FIG. 8 is a block diagram illustrating an embodiment where an off-host virtualizer is configured to dynamically map physical storage from within two different physical storage devices to a single virtual LUN.

[0020] FIG. 9 is a block diagram illustrating an embodiment where an off-host virtualizer is configured to dynamically map physical storage from within a single physical storage device to two virtual LUNs assigned to different hosts.

[0021] FIG. 10 is a block diagram illustrating an embodiment where an off-host virtualizer is configured to aggregate storage of a physical storage device into a logical volume and dynamically map the logical volume to a range of blocks of a virtual LUN.

[0022] FIG. 11 is a block diagram illustrating an embodiment where an off-host virtualizer is configured to dynamically map multiple logical volumes to a single virtual LUN.

[0023] FIG. 12 is a block diagram illustrating an embodiment where an off-host virtualizer is configured to aggregate storage from a physical storage device into two logical volumes, and to dynamically map each of the two logical volumes to a different virtual LUN.

[0024] FIG. 13 is a block diagram illustrating an embodiment employing multiple storage networks.

[0025] FIG. 14 is a block diagram illustrating an embodiment where an off-host virtualizer is configured to aggregate storage from two physical storage devices into a single logical volume.

[0026] FIG. 15 is a flow diagram illustrating aspects of the operation of a host and an off-host virtualizer according to one embodiment where the off-host virtualizer is configured to support dynamic physical LUN tunneling.

[0027] FIG. 16 is a flow diagram illustrating aspects of the operation of a host and an off-host virtualizer according to one embodiment, where off-host virtualizer 180 is configured to support dynamic volume tunneling.

[0028] FIG. 17 is a block diagram illustrating an embodiment where an off-host virtualizer comprises a virtualization switch.

12/21/2022 19:09:40 Page 6 of 52
SAMSUNG EX1002, Page 280 of 1031

[0029] FIG. 18 is a block diagram illustrating an embodiment where off-host virtualizer comprises a virtualization appliance.

[0030] FIG. 19 is a block diagram illustrating an exemplary host, according to one embodiment.

[0031] FIG. 20 is a block diagram illustrating a computer-accessible medium according to one embodiment.

[0032] While the invention is susceptible to various modifications and alternative forms, specific embodiments are shown by way of example in the drawings and are herein described in detail. It should be understood, however, that drawings and detailed description thereto are not intended to limit the invention to the particular form disclosed, but on the contrary, the invention is to cover all modifications, equivalents and alternatives falling within the spirit and scope of the present invention as defined by the appended claims.

DETAILED DESCRIPTION

[0033] FIG. 1a is a block diagram illustrating a computer system 100 according to one embodiment. System 100 includes a host 110 coupled to a physical block device 120 via an interconnect 130. Host 110 includes a traditional software storage stack 140A that may be used to perform I/O operations on physical block device 120 via interconnect 130.

[0034] Generally speaking, a physical block device 120 may comprise any hardware entity that provides a collection of linearly addressed data blocks that can be read or written. For example, in one embodiment a physical block device may be a single disk drive configured to present all of its sectors as an indexed array of blocks. In another embodiment the physical block device may be a disk array device, or a disk configured as part of a disk array device. It is contemplated that any suitable type of storage device may be configured as a block device, such as fixed or removable magnetic media drives (e.g., hard drives, floppy or Zip-based drives), writable or read-only optical media drives (e.g., CD or DVD), tape drives, solid-state mass storage devices, or any other type of storage device. The interconnect 130 may utilize any desired storage connection technology, such as various variants of the Small Computer System Interface (SCSI) protocol, Fiber Channel, Internet Protocol (IP), Internet SCSI (iSCSI), or a combination of such storage networking technologies. The software storage stack 140A may comprise layers of software within an operating system at host 110, and may be accessed by a client application to perform I/O (input/output) on a desired physical block device 120.

[0035] In the traditional storage software stack for block device access, a client application may initiate an I/O request, for example as a request to read a block of data at a specified offset within a file. The request may be received (e.g., in the form of a read() system call) at the file system layer 112, translated into a request to read a block within a particular device object (i.e., a software entity representing a storage device), and passed to the disk driver layer 114. The disk driver layer 114 may then select the targeted physical block device 120 corresponding to the disk device object, and send a request to an address at the targeted physical block device over the interconnect 130 using the interconnect-dependent I/O driver layer 116. For example, a host bus adapter (HBA) (such as a SCSI HBA) may be used to transfer the I/O request, formatted according to the appropriate storage protocol (e.g., SCSI), to a physical link of the interconnect (e.g., a SCSI bus). At the physical block device 120, an interconnect-dependent firmware layer 122 may receive the request, perform the desired physical I/O operation at the physical storage layer 124, and send the results back to the host over the interconnect. The results (e.g., the desired blocks of the file to be

SAMSUNG EX1002, Page 281 of 1031

read) may then be transferred through the various layers of storage stack 140A in reverse order (i.e., from the interconnect-dependent I/O driver to the file system) before being passed to the requesting client application.

[0036] In some operating systems, a hierarchical scheme may be used for addressing storage within physical block devices 120. For example, an operating system may employ a four-level hierarchical addressing scheme of the form <*"hba", "bus", "target", "lun"*> for SCSI devices, including a SCSI HBA identifier (*"hba"*), a SCSI bus identifier (*"bus"*), a SCSI target identifier (*"target"*), and a logical unit identifier (*"lun"*), and may be configured to populate a device database with addresses for available SCSI devices during boot. Host 110 may include multiple SCSI HBAs, and a different SCSI adapter identifier may be used for each HBA. The SCSI adapter identifiers may be numbers issued by the operating system kernel, for example based on the physical placement of the HBA cards relative to each other (i.e., based on slot numbers used for the adapter cards). Each HBA may control one or more SCSI buses, and a unique SCSI bus number may be used to identify each SCSI bus within an HBA. During system initialization, or in response to special configuration commands, the HBA may be configured to probe each bus to identify the SCSI devices currently attached to the bus. Depending on the version of the SCSI protocol in use, the number of devices (such as disks or disk arrays) that may be attached on a SCSI bus may be limited, e.g., to 15 devices excluding the HBA itself. SCSI devices that may initiate I/O operations, such as the HBA, are termed SCSI initiators, while devices where the physical I/O may be performed are called SCSI targets. Each target on the SCSI bus may identify itself to the HBA in response to the probe. In addition, each target device may also accommodate up to a protocol-specific maximum number of "logical units" (LUNs) representing independently addressable units of physical storage within the target device, and may inform the HBA of the logical unit identifiers. A target device may contain a single LUN (e.g., a LUN may represent an entire disk or even a disk array) in some embodiments. The SCSI device configuration information, such as the target device identifiers and LUN identifiers may be passed to the disk driver layer 114 by the HBAs. When issuing an I/O request, disk driver layer 114 may utilize the hierarchical SCSI address described above.

[0037] When accessing a LUN, disk driver layer 114 may expect to see OS-specific metadata at certain specific locations within the LUN. For example, in many operating systems, the disk driver layer 114 may be responsible for implementing logical partitioning (i.e., subdividing the space within a physical disk into partitions, where each partition may be used for a smaller file system). Metadata describing the layout of a partition (e.g., a starting block offset for the partition within the LUN, and the length of a partition) may be stored in an operating-system dependent format, and in an operating system-dependent location, such as in a header or a trailer, within a LUN. In the Solaris™ operating system from Sun Microsystems, for example, a virtual table of contents (VTOC) structure may be located in the first partition of a disk volume, and a copy of the VTOC may also be located in the last two cylinders of the volume. In addition, the operating system metadata may include cylinder alignment and/or cylinder size information, as well as boot code if the volume is bootable. Operating system metadata for various versions of Microsoft Windows™ may include a "magic number" (a special number or numbers that the operating system expects to find, usually at or near the start of a disk), subdisk layout information, etc. If the disk driver layer 114 does not find the metadata in the expected location and in the expected format, the disk driver layer may not be able to perform I/O operations at the LUN.

[0038] The relatively simple traditional storage software stack 140A has been enhanced over time to help provide advanced storage features, most significantly by introducing block virtualization layers. In general, block

12/21/2022 19:09:40 Page 8 of 52
SAMSUNG EX1002, Page 282 of 1031

virtualization refers to a process of creating or aggregating logical or virtual block devices out of one or more underlying physical or logical block devices, and making the virtual block devices accessible to block device consumers for storage operations. For example, in one embodiment of block virtualization, storage within multiple physical block devices, e.g. in a fiber channel storage area network (SAN), may be aggregated and presented to a host as a single virtual storage device such as a virtual LUN (VLUN), as described below in further detail. In another embodiment, one or more layers of software may rearrange blocks from one or more block devices, such as disks, and add various kinds of functions. The resulting rearranged collection of blocks may then be presented to a storage consumer, such as an application or a file system, as one or more aggregated devices with the appearance of one or more basic disk drives. That is, the more complex structure resulting from rearranging blocks and adding functionality may be presented as if it were one or more simple arrays of blocks, or logical block devices. In some embodiments, multiple layers of virtualization may be implemented. In such embodiments, one or more block devices may be mapped into a particular virtualized block device, which may be in turn mapped into still another virtualized block device, allowing complex storage functions to be implemented with simple block devices. As used herein, the term "logical volume" refers to a virtualized block device that may be presented directly for use by a file system, database, or other applications that can directly use block devices. Further details on block virtualization, and advanced storage features supported by block virtualization, are provided below.

[0039] Block virtualization may be implemented at various places within a storage stack and the associated storage environment, in both hardware and software. For example, a block virtualization layer in the form of a volume manager, such as the VERITAS Volume Manager™ from VERITAS Software Corporation, may be added between the disk driver layer 114 and the file system layer 112. In some storage environments, virtualization functionality may be added to host bus adapters, i.e., in a layer between the interconnect-dependent I/O driver layer 116 and interconnect 130. Block virtualization may also be performed outside the host 110, e.g., in a virtualization appliance or a virtualizing switch, which may form part of the interconnect 130. Such external devices providing block virtualization (i.e., devices that are not incorporated within host 110) may be termed off-host virtualizers or off-host virtualization controllers. In some storage environments, block virtualization functionality may be implemented by an off-host virtualizer in cooperation with a host-based virtualizer. That is, some block virtualization functionality may be performed off-host, and other block virtualization features may be implemented at the host. In another embodiment, multiple devices external to the host 110, such as two or more virtualization switches, two or more virtualization appliances, or a combination of virtualization switches and virtualization appliances, may cooperate to provide block virtualization functionality; that is, the off-host virtualizer may include multiple cooperating devices.

[0040] While additional layers may be added to the storage software stack 140A, it is generally difficult to remove or completely bypass existing storage software layers of operating systems. Therefore, off-host virtualizers may typically be implemented in a manner that allows the existing storage software layers to continue to operate, even when the storage devices being presented to the operating system are virtual rather than physical, and remote rather than local. For example, because disk driver layer 114 expects to deal with SCSI LUNs when performing I/O operations, an off-host virtualizer may present a virtualized storage device to the disk driver layer as a virtual LUN. That is, as described below in further detail, on off-host virtualizer may encapsulate, or emulate the metadata for, a LUN when providing a host 110 access to a virtualized storage device. In addition, as also described below, one or

12/21/2022 19:09:40 Page 9 of 52
SAMSUNG EX1002, Page 283 of 1031

more software modules or layers may be added to storage stack 140A to support additional forms of virtualization using virtual LUNs.

[0041] FIG. 1b is a block diagram illustrating an embodiment of system 100 configured to utilize off-host block virtualization. As shown, the system may include an off-host virtualizer 180, such as a virtualization switch or a virtualization appliance, which may be included within interconnect 130 linking host 110 to physical block device 120. Host 110 may comprise an enhanced storage software stack 140B, which may include an intermediate driver layer 113 between the disk driver layer 114 and file system layer 112. In one embodiment, off-host virtualizer 180 may be configured to map storage within physical block device 120, or multiple physical block devices 120, into a virtual storage device (e.g., a virtual LUN or VLUN), and present the virtual storage device to host 110.

[0042] The back-end physical block device 120 that is mapped to a virtual LUN may be termed a "physical LUN (PLUN)" in the subsequent description. In one embodiment, off-host virtualizer 180 may be configured to map storage from a back-end physical LUN directly to a VLUN, without any additional virtualization (i.e., without creating a logical volume). Such a technique of mapping a PLUN to a VLUN may be termed "PLUN tunneling" in the subsequent description. In another embodiment, off-host virtualizer 180 may be configured to aggregate storage within one or more physical block devices 120 as one or more logical volumes, and map the logical volumes within the address space of a VLUN presented to host 110. The terms "volume tunneling" or "logical volume tunneling" may be used herein to refer to the technique of mapping a logical volume through a VLUN. Off-host virtualizer 180 may further be configured to provide intermediate driver layer 130 with metadata or configuration information on the tunneled logical volumes, allowing intermediate driver layer 130 to locate and perform I/O operations on the logical volumes located within the virtual LUN on behalf of clients such as file system layer 112 or other applications. In some embodiments, an aliasing technique such as symbolic links may be used to associate a name with a tunneled logical volume, allowing file systems, applications and system administrators to refer to the tunneled logical volume using a conventional or recommended naming scheme. In one embodiment, the selection of a name may also be coordinated between the host 110 and the off-host virtualizer 180, e.g., using a field in an extended SCSI mode page, or through an external virtualization management infrastructure including one or more additional devices and/or hosts. File system layer 112 and applications (such as database management systems) configured to utilize intermediate driver layer 113 and lower layers of storage stack 140B may be termed "virtual storage clients" or "virtual storage consumers" herein. While off-host virtualizer 180 is shown within interconnect 130 in the embodiment depicted in FIG. 1b, it is noted that in other embodiments, off-host virtualization may also be provided within physical block device 120 (e.g., by a virtualization layer between physical storage layer 124 and firmware layer 122), or at another device outside interconnect 130. As described below in further detail, in some embodiments a number of independently configurable interconnects may be employed to link off-host virtualizer 180 to hosts 110 and back-end storage devices 120: e.g., one or more front-end interconnects 130 or storage area network (SAN) fabrics may link off-host virtualizer 180 to hosts 110, while one or more back-end interconnects 130 may link off-host virtualizer to back-end storage devices.

[0043] As described above, in some embodiments, disk driver layer 114 may expect certain operating system-specific metadata to be present at operating-system specific locations or offsets within a LUN. When presenting a virtual LUN to a host 110, therefore, in such embodiments off-host virtualizer 180 may logically insert the expected metadata at the expected locations. FIG. 2a is a block diagram illustrating the addition of operating-system specific

12/21/2022 19:09:40 Page 10 of 52
SAMSUNG EX1002, Page 284 of 1031

metadata to a virtual LUN 210 encapsulating a source volume 205, according to one embodiment. As shown, the source volume 205 consists of N blocks, numbered 0 through (N-1). The virtual LUN 210 may include two regions of inserted metadata: a header 215 containing H blocks of metadata, and a trailer 225 including T blocks of metadata. Between the header 215 and the trailer 225, blocks 220 of the virtual LUN 210 may be mapped to the source volume 205, thereby making the virtual LUN 210 a total of (H+N+T) blocks long (i.e., the virtual LUN may contain blocks numbered 0 through (H+N+T-1)). Operating-system specific metadata included in header 215 and/or trailer 225 may be used by disk driver layer 114 to recognize the virtual LUN 210 as an addressable storage device. An "addressable storage device", as used herein, is a storage device whose blocks can be accessed (e.g., from a device driver such as a disk driver) using an address including a device identifier (such as a logical unit identifier) and an offset within the device. In some embodiments, additional configuration information or logical volume metadata may also be included within header 215 and/or trailer 225. The lengths of header 215 and trailer 225, as well as the format and content of the metadata, may vary with the operating system in use at host 110. It is noted that in some embodiments, the metadata may require only a header 215, or only a trailer 225, rather than both a header and a trailer; and in other embodiments, the metadata may be stored at any arbitrary offset within the LUN.

[0044] Off-host virtualizer 180 may be configured to customize the generated operating system metadata (e.g., in header 215 and/or trailer 225) based on the specific requirements of the operating system in use at host 110. The requirements imposed by different operating systems may differ in the type of information to be included in the metadata (e.g., whether the metadata includes a partition table), the format in which the information is maintained (e.g., the units in which a partition or volume size is specified such as kilobytes or 512-byte blocks, whether offsets are expressed in hexadecimal or decimal numerals, etc.), the location within the LUN where the metadata is to be found, etc. In some embodiments, the requirements may be sent from the host 110 to the off-host-virtualizer 180. For example, in one embodiment a virtualization management infrastructure may be implemented using one or more devices and/or additional hosts (i.e., using devices and/or hosts other than host 110 and off-host virtualizer 180). A device or host of the virtualization management infrastructure may be configured to communicate with the host 110, obtain the metadata requirements for the operating system from the host 110, and provide the requirements to the off-host virtualizer 180. In another embodiment, host 110 may be configured to provide the operating system requirements to off-host virtualizer, for example using an extension of a storage command (such as an extended SCSI request). The storage command and/or the extension may be vendor unique (i.e., a storage command may need to be extended in different ways for different hardware storage vendors) in some embodiments. In one embodiment, only an identity of the operating system in use at the host 110 (e.g., the operating system name and version number) may be provided to the off-host virtualizer 180, and the off-host virtualizer may be configured to obtain the details of the operating system requirements using the operating system identity (e.g., from a database maintained at the off-host virtualizer 180 or from an external database). Operating system identity and/or metadata requirements for a host 110 may also be specified to off-host virtualizer 180 manually in some embodiments, e.g., by a system administrator using a graphical user interface or a command-line interface. In some embodiments, a common metadata format shared by multiple operating systems may be used.

[0045] The metadata inserted within virtual LUN 210 may be stored in persistent storage, e.g., within some blocks of physical block device 120 or at storage within off-host virtualizer 180, in some embodiments, and logically concatenated with the mapped blocks 220. In other embodiments, the metadata may be maintained in non-

12/21/2022 19:09:40 Page 11 of 52
SAMSUNG EX1002, Page 285 of 1031

persistent storage (e.g., within a memory at off-host virtualizer 180) and/or generated dynamically whenever a host 110 accesses the virtual LUN 210. In some embodiments, the metadata may be generated by an external agent other than off-host virtualizer 180. The external agent may be capable of emulating metadata in a variety of formats for different operating systems, including operating systems that may not have been known when the off-host virtualizer 180 was deployed. In one embodiment, off-host virtualizer 180 may be configured to support more than one operating system; i.e., off-host virtualizer may logically insert metadata blocks corresponding to any one of a number of different operating systems when presenting virtual LUN 210 to a host 110, thereby allowing hosts with different operating systems to share access to a particular storage device 120.

[0046] As stated earlier, in some embodiments off-host virtualizer 180 may be configured to generate operating system metadata including partition layout information. FIG. 2b is a block diagram illustrating one embodiment where off-host virtualizer 180 is configured to include a partition table 255 within the generated operating system metadata header 215. The partition table 255 may provide layout or mapping information describing the offset within the VLUN 210 at which the blocks mapped to a logical partition of a physical storage device 250 may be located. In the illustrated embodiment, the layout table may contain three entries corresponding to Partition 1, Partition 2 and Partition 3 of physical storage device 250, respectively.

[0047] Off-host virtualizer may also be configured to map one or more logical volumes to the address space of a virtual LUN in some embodiments. FIG. 2c is a block diagram illustrating an embodiment where off-host virtualizer 180 is configured to aggregate storage within physical storage devices 250A and 250B into a logical volume 260, and to map logical volume 260 to blocks 261 of a virtual LUN 210. In addition to generating operating system metadata as described above, off-host virtualizer 180 may also be configured to generate metadata specific to the logical volume in such embodiments. Such logical volume metadata 263 may include information (such as a starting offset, volume length, virtualization layer information such as number of mirrors, etc.) allowing intermediate driver layer 113 to perform I/O operations on the logical volume 260. It is noted that in one embodiment, logical volume metadata may be included within header 215 or trailer 225, i.e., logical volume metadata intended for use by intermediate driver 113 may be co-located with operating system metadata intended for use by disk driver layer 114. While the logical volume 260 illustrated in FIG. 2c is backed by two physical storage devices 250A and 250B, in general, storage from any number of physical storage devices 250 (including from a single physical storage device) may be aggregated into a logical volume 260.

[0048] FIG. 3 is a block diagram illustrating an embodiment including an off-host virtualizer 180 configured to create a plurality of virtual LUNs (VLUNs) 210. In the illustrated embodiment, hosts 110A, 110B and 110C (collectively, hosts 110) may be coupled to off-host virtualizer 180 via interconnect 130A, and off-host virtualizer 180 may be coupled to storage devices 340A, 340B and 340C (collectively, storage devices 340) via interconnect 130B. Storage devices 340 may include physical block devices 120 as well as virtual block devices (e.g., in embodiments employing multiple layers of virtualization, as described below). Hosts 110A and 110B may utilize Operating System A, while host 110C may utilize Operating System B; that is, in general, each host among hosts 110 may support any of a number of operating systems. As shown, off-host virtualizer 180 may be configured to generate metadata for a number of VLUNs 210A-210E. VLUN 210A may be assigned to host 110A (i.e., the operating system metadata generated for VLUN 210A may be made accessible to host 110A, allowing host 110A to detect an existence of VLUN 210A as an addressable storage device). VLUN 210B may be assigned to host 110B,

12/21/2022 19:09:40 Page 12 of 52
SAMSUNG EX1002, Page 286 of 1031

while VLUNs 210C, 210D and 210E may all be assigned to host 110C. Storage within storage device 340A may be mapped to both VLUN 210A and VLUN 210E, as indicated by the dotted arrows in FIG. 3. Thus, each of two or more hosts with different operating systems (e.g., host 110A and host 110C) may be given access to the same storage device (e.g., 340A) using different VLUNs (e.g., 210A and 210E) in the illustrated embodiment. In general, off-host virtualizer 180 may be configured to support any desired total number of VLUNs 210, to assign any desired VLUNs to a given host, and to map storage from any desired combination of storage devices 340 to a given VLUN. It is noted that certain operating systems may impose a limit on the total number of LUNs (or VLUNs) that may be visible to a host 110, and/or the size of a given LUN or VLUN, and that off-host virtualizer may be configured to comply with such limits in some embodiments.

[0049] FIG. 4 is a flow diagram illustrating aspects of the operation of a host 110 and an off-host virtualizer 180 configured to generate operating system metadata as described above. Off-host virtualizer 180 may be configured to receive operating system metadata requirements (block 410) from host 110, e.g., using one of the techniques described above, such as an extended SCSI command. The off-host virtualizer 180 may then generate operating system metadata (block 420) for a virtual storage device such as a VLUN 210 in accordance with the requirements, and make the operating system metadata accessible to the host 110 (block 430). The metadata may be made accessible to host 110 via a message sent by off-host virtualizer 180 over interconnect 130A in some embodiments. In other embodiments, host 110A may be configured to send a query to off-host virtualizer 180, requesting information on the VLUNs assigned to the host 110, and the metadata may be provided in response to the query. A first layer of the storage software stack (e.g., disk driver layer 114) at host 110 may be configured to use the metadata to detect the existence of the virtual storage device as an addressable storage device (block 440). It is noted that in some embodiments, where for example all the hosts 110 in the storage environment are configured to use the same operating system, the operations corresponding to block 410 may be omitted, i.e., there may be no need to provide operating system-specific requirements to an off-host virtualizer 180 in homogeneous environments where a single operating system is in use at all the hosts 110.

[0050] FIG. 5 is a flow diagram illustrating an embodiment where an off-host virtualizer 180 is configured to aggregate storage in one or more physical storage devices into a logical volume 260, and to map the logical volume to a VLUN 210. As in FIG. 4, operating system metadata requirements may be received by off-host virtualizer 180 (block 510), and operating system metadata for a virtual storage device or VLUN 210 may be generated by off-host virtualizer in accordance with the requirements (block 520). Off-host virtualizer may be configured to aggregate storage within one or more physical storage devices 340 into a logical volume 260 (block 530) and to map the logical volume into the address space of the virtual storage device or VLUN (block 540). The operating system metadata, as well as additional metadata specific to the logical volume, may be provided to a host 110 (block 550). A first layer of a software storage stack at the host (such as disk driver layer 114) may be configured to use the operating system metadata to detect the existence of the virtual storage device (block 560), while a second layer (such as intermediate driver 113) may be configured to use the logical volume metadata to access and initiate I/O operations on the logical volume 260 (block 570).

[0051] In some embodiments, the basic techniques of PLUN tunneling and volume tunneling described above may be extended to allow dynamic association of PLUNs and/or logical volumes with VLUNs. While logical volumes such as logical volume 260 may typically be created and dynamically reconfigured (e.g., grown or shrunk,

12/21/2022 19:09:40 Page 13 of 52
SAMSUNG EX1002, Page 287 of 1031

imported to hosts 110 or exported from hosts 110) efficiently, similar configuration operations on LUNs (i.e., physical and/or virtual LUNs) may typically be fairly slow. Some LUN reconfiguration operations may be at least partially asynchronous, and may have unbounded completion times and/or ambiguous failure states. On many operating systems, LUN reconfiguration may only be completed after a system reboot; for example, a newly created physical or virtual LUN may not be detected by the operating system without a reboot. In order to be able to flexibly map logical volumes to virtual LUNs, while avoiding the delays and problems associated with LUN reconfigurations, therefore, it may be advisable to generate unmapped virtual LUNs (i.e., to create operating system metadata for virtual LUNs that are not initially mapped to any physical LUNs or to logical volumes) and pre-assign the unmapped virtual LUNs to hosts 110 as part of an initialization process. The initialization process may be completed prior to performing storage operations on the virtual LUNs on behalf of applications. During the initialization process (which may include a reboot of the system in some embodiments) the layers of the software storage stack 140B may be configured to detect the existence of the virtual LUNs as addressable storage devices. Subsequent to the initialization, off-host virtualizer 180 may dynamically map physical LUNs and/or logical volumes to the virtual LUNs (e.g., by modifying portions of the operating system metadata), as described below in further detail. The term "dynamic mapping", as used herein, refers to a mapping of a virtual storage device (such as a VLUN) that is performed by modifying one or more blocks of metadata, and/or by communicating via one or more messages to a host 110, without requiring a reboot of the host 110 to which the virtual storage device is presented.

[0052] FIG. 6 is a block diagram illustrating an example of an unmapped virtual LUN 230 according to one embodiment. As shown, the unmapped virtual LUN 230 may include an operating system metadata header 215 and an operating system metadata trailer 225, as well as a region of unmapped blocks 235. In some embodiments, the size of the region of unmapped blocks (X blocks in the depicted example) may be set to a maximum permissible LUN or maximum logical volume size supported by an operating system, so that any subsequent mapping of a logical volume or physical LUN to the virtual LUN does not require an expansion of the size of the virtual LUN. In one alternative embodiment, the unmapped virtual LUN may consist of only the emulated metadata (e.g., header 215 and/or trailer 225), and the size of the virtual LUN may be increased when the volume or physical LUN is dynamically mapped. In such embodiments, disk driver layer 114 may have to modify some of its internal data structures when the virtual LUN is expanded, and may have to re-read the emulated metadata in order to do so. Off-host virtualizer 180 may be configured to send a metadata change notification message to disk driver layer 114 in order to trigger the re-reading of the metadata.

[0053] FIG. 7 is a block diagram illustrating an embodiment including an off-host virtualizer 180 configured to create a plurality of unmapped virtual LUNs (VLUNs) 230. As shown, more than one unmapped VLUN may be associated with a single host 110. For example, off-host virtualizer 180 may assign unmapped VLUNs 230A and 230B to host 110A, and unmapped VLUNs 230C, 230D and 230E to host 110B. In some embodiments, multiple VLUNs may be associated with a given host to allow for isolation of storage used for different applications, or to allow access to storage beyond the maximum allowable LUN size supported in the system. Off-host virtualizer 180 may be configured to dynamically map physical and/or virtual storage from storage devices 340 to the unmapped virtual LUNs 230. Hosts 110A and 110B may be configured to use different operating systems in some embodiments, and may utilize the same operating system in other embodiments. After the physical or virtual storage has been dynamically mapped, the previously unmapped VLUN 230 may provide the same general functionality as

12/21/2022 19:09:40 Page 14 of 52
SAMSUNG EX1002, Page 288 of 1031

described previously for VLUNs 210 provided in basic (i.e., non-dynamic) PLUN tunneling and basic volume tunneling. A number of different techniques for dynamic mapping of physical storage and logical volumes to VLUNs are described below. It is noted that each such technique (e.g., a mapping from two PLUNs to a single VLUN) may also be implemented using basic PLUN tunneling or basic volume tunneling in some embodiments, i.e., without starting with unmapped VLUNs pre-assigned to hosts 110.

[0054] As described earlier, operating system-specific metadata (e.g., in header 215 and/or trailer 225 of FIG. 2b) generated by off-host virtualizer 180 may allow disk driver layer 114 at a host 110 to detect the existence of an unmapped virtual LUN 230 as an addressable storage device. After VLUN 230 has been recognized by disk driver layer 114, a block at any offset within the VLUN address space may be accessed by the disk driver layer 114, and thus by any other layer above the disk driver layer. For example, intermediate driver layer 113 may be configured to communicate with off-host virtualizer 180 by reading from, and/or writing to, a designated set of blocks emulated within VLUN 230. Such designated blocks may provide a mechanism for off-host virtualizer 180 to provide intermediate driver layer 113 with configuration information associated with logical volumes or physical LUNs mapped to VLUN 230 in some embodiments.

[0055] In one embodiment, off-host virtualizer 180 may be configured to dynamically map storage from a back-end physical LUN directly to an unmapped VLUN 230, without any additional virtualization (i.e., without creating a logical volume). Such a technique of dynamically mapping a PLUN to a VLUN 230 may be termed "dynamic PLUN tunneling" (in contrast to the basic PLUN tunneling described above, where unmapped VLUNs may not be used). Each PLUN may be mapped to a corresponding VLUN 230 (i.e., a 1-to-1 mapping of PLUNs to VLUNs may be implemented by off-host virtualizer 180) in some embodiments. In other embodiments, as described below in conjunction with the description of FIG. 8, storage from multiple PLUNs may be mapped into subranges of a given VLUN 230. In general, both basic and dynamic PLUN tunneling may allow the off-host virtualizer 180 to act as an isolation layer between VLUNs 230 (the storage entities directly accessible to hosts 110) and back-end PLUNs, allowing the off-host virtualizer to hide details related to physical storage protocol implementation from the hosts. In one implementation, for example, the back-end PLUNs may implement a different version of a storage protocol (e.g., SCSI-3) than the version seen by hosts 100 (e.g., SCSI-2), and the off-host virtualizer may provide any needed translation between the two versions. In another implementation, off-host virtualizer 180 may be configured to implement a cooperative access control mechanism for the back-end PLUNs, and the details of the mechanism may remain hidden from the hosts 110.

[0056] In addition, off-host virtualizer 180 may also be configured to increase the level of data sharing using PLUN tunneling (i.e., either basic PLUN tunneling or dynamic PLUN tunneling). Disk array devices often impose limits on the total number of concurrent "logins", i.e., the total number of entities that may access a given disk array device. In a storage environment employing PLUN tunneling for disk arrays (i.e., where the PLUNs are disk array devices), off-host virtualizers 180 may allow multiple hosts to access the disk arrays through a single login. That is, for example, multiple hosts 110 may log in to the off-host virtualizer 180, while the off-host virtualizer may log in to a disk array PLUN once on behalf of the multiple hosts 110. Off-host virtualizer 180 may then pass on I/O requests from the multiple hosts 110 to the disk array PLUN using a single login. The number of logins (i.e., distinct entities logged in) as seen by a disk array PLUN may thereby be reduced as a result of PLUN tunneling, without reducing the number of hosts 110 from which I/O operations targeted at the disk array PLUN may be initiated. The total

12/21/2022 19:09:40 Page 15 of 52
SAMSUNG EX1002, Page 289 of 1031

number of hosts 110 that may access storage at a single disk array PLUN with login count restrictions may thereby be increased, thus increasing the overall level of data sharing.

[0057] FIG. 8 is a block diagram illustrating an embodiment where an off-host virtualizer 180 is configured to dynamically map physical storage from within two different physical storage devices 340A and 340B to a single VLUN 230B. That is, off-host virtualizer 180 may be configured to map a first range of physical storage from device 340A into a first region of mapped blocks 821A within VLUN 230B, and map a second range of physical storage from device 340B into a second region of mapped blocks 821B within VLUN 230B. The first and second ranges of physical storage may each represent a respective PLUN, such as a disk array, or a respective subset of a PLUN. Configuration information indicating the offsets within VLUN 230B at which mapped blocks 821A and 821B are located may be provided by off-host virtualizer 180 to intermediate driver layer 113 using a variety of mechanisms in different embodiments. For example, in one embodiment, off-host virtualizer 180 may write the configuration information to a designated set of blocks within VLUN 230, and intermediate driver layer 113 may be configured to read the designated set of blocks, as described above. In another embodiment, off-host virtualizer 180 may send a message containing the configuration information to host 110A, either directly (over interconnect 350A or another network) or through an intermediate coordination server. In yet another embodiment, the configuration information may be supplied within a special SCSI mode page (i.e., intermediate driver layer 113 may be configured to read a special SCSI mode page containing configuration information updated by off-host virtualizer 180). Combinations of these techniques may be used in some embodiments: for example, in one embodiment off-host virtualizer 180 may send a message to intermediate driver layer 113 requesting that intermediate driver layer read a special SCSI mode page containing the configuration information.

[0058] FIG. 9 is a block diagram illustrating an embodiment where off-host virtualizer 180 is configured to dynamically map physical storage from within a single physical storage device 340A to two VLUNs 230 assigned to different hosts 110A and 110B. As shown, a first range of physical storage 955A of physical storage device 340A may be mapped to a first range of mapped blocks 821A within VLUN 230B assigned to host 110A. A second range of physical storage 955B of the same physical storage device 340A may be mapped to a second range of mapped blocks 821C of VLUN 230E assigned to host 110B. In addition, in some embodiments, off-host virtualizer 180 may be configured to prevent unauthorized access to physical storage range 955A from host 110B, and to prevent unauthorized access to physical storage 955B from host 110A. Thus, in addition to allowing access to a single physical storage device 340A from multiple hosts 110, off-host virtualizer 180 may also be configured to provide security for each range of physical storage 955A and 955B, e.g., in accordance with a specified security protocol. In one embodiment, for example, the security protocol may allow I/O operations to a given VLUN 230 (and to its backing physical storage) from only a single host 110. Off-host virtualizer 180 may be configured to maintain access rights information for the hosts 110 and VLUNs 230 in some embodiments, while in other embodiments security tokens may be provided to each host 110 indicating the specific VLUNs to which access from the host is allowed, and the security tokens may be included with I/O requests.

[0059] In some embodiments off-host virtualizer 180 may be configured to aggregate physical storage into a logical volume, and dynamically map the logical volume to an address range within a VLUN 230. Such a technique of dynamically mapping a logical volume to a VLUN 230 may be termed "dynamic volume tunneling" (in contrast to the basic volume tunneling described above, where unmapped VLUNs may not be used).FIG. 10 is a block

12/21/2022 19:09:40 Page 16 of 52
SAMSUNG EX1002, Page 290 of 1031

diagram illustrating an embodiment where off-host virtualizer 180 is configured to aggregate storage 1055A of physical storage device 340A into a logical volume 1060A, and dynamically map logical volume 1060A to a range of blocks (designated as mapped volume 1065A in FIG. 10) of VLUN 230B. As in the case of PLUN tunneling, configuration information or metadata associated with the tunneled logical volume 1060A may be provided to intermediate driver layer 113 using any of a variety of mechanisms, such as an extended SCSI mode page, emulated virtual blocks within VLUN 230A, and/or direct or indirect messages dent from off-host virtualizer 180 to host 110A. While logical volume 1060A is shown as being backed by a portion of a single physical storage device 340A in the depicted embodiment, in other embodiments logical volume 1060A may be aggregated from all the storage within a single physical storage device, or from storage of two or more physical devices. In some embodiments employing multiple layers of virtualization, logical volume 1060A may itself be aggregated from other logical storage devices rather than directly from physical storage devices. In one embodiment, each host 110 (i.e., host 110B in addition to host 110A) may be provided access to logical volume 1060A via a separate VLUN, while in another embodiment different sets of logical volumes may be presented to different hosts 110.

[0060] FIG. 11 is a block diagram illustrating an embodiment where off-host virtualizer 180 is configured to dynamically map multiple logical volumes to a single VLUN 230. As shown, off-host virtualizer 180 may be configured to aggregate storage range 1155C from physical storage device 340A, and physical storage range 1155B from physical storage device 340B, into a logical volume 1160A, and map logical volume 1160A to a first mapped volume region 1165A of VLUN 230B. In addition, off-host virtualizer 180 may also aggregate physical storage range 1155B from physical storage device 340A into a second logical volume 1160B, and map logical volume 760B to a second mapped volume region 1165B of VLUN 230B. In general, off-host virtualizer 180 may aggregate any suitable selection of physical storage blocks from one or more physical storage devices 340 into one or more logical volumes, and map the logical volumes to one or more of the pre-generated unmapped VLUNs 230.

[0061] FIG. 12 is a block diagram illustrating another embodiment, where off-host virtualizer 180 is configured to aggregate storage from physical storage device 340A into logical volumes 1260A and 1260B, and to dynamically map each of the two logical volumes to a different VLUN 230. For example, as shown, logical volume 1260A may be mapped to a first address range within VLUN 230B, accessible from host 110A, while logical volume 1260B may be mapped to a second address range within VLUN 230E, accessible from host 110B. Off-host virtualizer 180 may further be configured to implement a security protocol to prevent unauthorized access and/or data corruption, similar to the security protocol described above for PLUN tunneling. Off-host virtualizer 180 may implement the security protocol at the logical volume level: that is, off-host virtualizer 180 may prevent unauthorized access to logical volumes 1260A (e.g., from host 110B) and 1260B (e.g., from host 110A) whose data may be stored within a single physical storage device 340A. In one embodiment, off-host virtualizer 180 may be configured to maintain access rights information for logical volumes 1260 to which each host 110 has been granted access. In other embodiments security tokens may be provided to each host 110 (e.g., by off-host virtualizer 180, or by an external security server) indicating the specific logical volumes 1260 to which access from the host is allowed, and the security tokens may be included with I/O requests.

[0062] Many storage environments utilize storage area networks (SANs), such as fibre channel fabrics, to access physical storage devices. SAN fabric reconfiguration (e.g., to provide access to a particular PLUN or logical volume from a particular host that did not previously have access to the desired PLUN or logical volume), which

12/21/2022 19:09:40 Page 17 of 52
SAMSUNG EX1002, Page 291 of 1031

may require switch reconfigurations, recabling, rebooting, etc., may typically be fairly complex and error-prone. The techniques of dynamic PLUN tunneling and dynamic volume tunneling, described above, may allow a simplification of SAN reconfiguration operations. By associating pre-generated, unmapped VLUNs to hosts, and mapping PLUNs and logical volumes to VLUNs dynamically as needed, many reconfiguration operations may require only a change of a mapping table at a switch, and a recognition of new metadata by intermediate driver layer 113. Storage devices may be more easily shared across multiple hosts 110, or logically transferred from one host to another, using PLUN tunneling and/or volume tunneling. Allocation and/or provisioning of storage, e.g., from a pool maintained by a coordinating storage allocator, may also be simplified.

[0063] In addition to simplifying SAN configuration changes, PLUN tunneling and volume tunneling may also support storage interconnection across independently configured storage networks (e.g., interconnection across multiple fiber channel fabrics). FIG. 13 is a block diagram illustrating an embodiment employing multiple storage networks. As shown, off-host virtualizer 180 may be configured to access physical storage device 340A via a first storage network 1310A, and to access physical storage device 340B via a second storage network 1310B. Off-host virtualizer 180 may aggregate storage from physical storage device 340A into logical volume 1360A, and map logical volume 1360A to VLUN 230B. Similarly, off-host virtualizer 180 may aggregate storage from physical storage device 340B into logical volume 1360B, and map logical volume 1360B to VLUN 230E. Host 110A may be configured to access VLUN 230A via a third storage network 1310C, and to access VLUN 230B via a fourth storage network 1310D.

[0064] Each storage network 1310 (i.e., storage network 1310A, 1310B, 1310C, or 1310D) may be independently configurable: that is, a reconfiguration operation performed within a given storage network 1310 may not affect any other storage network 1310. A failure or a misconfiguration within a given storage network 1310 may also not affect any other independent storage network 1310. In some embodiments, hosts 110 may include multiple HBAs, allowing each host to access multiple independent storage networks. For example, host 110A may include two HBAs in the embodiment depicted in FIG. 13, with the first HBA allowing access to storage network 1310C, and the second HBA to storage network 1310D. In such an embodiment, host 110A may be provided full connectivity to back-end physical storage devices 340, while still maintaining the advantages of configuration isolation. While FIG. 13 depicts the use of multiple independent storage networks in conjunction with volume tunneling, in other embodiments multiple independent storage networks may also be used with PLUN tunneling, or with a combination of PLUN and volume tunneling. In addition, it is noted that in some embodiments, the use of independent storage networks 1310 may be asymmetric: e.g., in one embodiment, multiple independent storage networks 1310 may be used for front-end connections (i.e., between off-host virtualizer 180 and hosts 110), while only a single storage network may be used for back-end connections (i.e., between off-host virtualizer 180 and physical storage devices 340). Any desired interconnection technology and/or protocol may be used to implement storage networks 1310, such as fiber channel, IP-based protocols, etc. For example, in one embodiment each storage network 1310 may be a fibre channel fabric. In another embodiment, the interconnect technology or protocol used within a first storage network 1310 may differ from the interconnect technology or protocol used within a second storage network 1310.

[0065] In one embodiment, volume tunneling may also allow maximum LUN size limitations to be overcome. For example, the SCSI protocol may be configured to use a 32-bit unsigned integer as a LUN block address, thereby

12/21/2022 19:09:40 Page 18 of 52
SAMSUNG EX1002, Page 292 of 1031

limiting the maximum amount of storage that can be accessed at a single LUN to 2 terabytes (for 512-byte blocks) or 32 terabytes (for 8-kilobyte blocks). Volume tunneling may allow an intermediate driver layer 113 to access storage from multiple physical LUNs as a volume mapped to a single VLUN, thereby overcoming the maximum LUN size limitation. FIG. 14 is a block diagram illustrating an embodiment where off-host virtualizer 180 may be configured to aggregate storage from two physical storage devices 340A and 340B into a single logical volume 1460A, where the size of the volume 1460A exceeds the allowed maximum LUN size supported by the storage protocol in use at storage devices 340. Off-host virtualizer 180 may further be configured to dynamically map logical volume 1460A to VLUN 230B, and to provide logical volume metadata to intermediate driver layer 113 at host 110A. The logical volume metadata may include sufficient information for intermediate driver layer 113 to access a larger address space within VLUN 230B than the maximum allowed LUN size.

[0066] FIG. 15 is a flow diagram illustrating aspects of the operation of host 110 and off-host virtualizer 180 according to one embodiment, where off-host virtualizer 180 is configured to support dynamic PLUN tunneling. After receiving operating system metadata requirements (block 1505), off-host virtualizer 180 may be configured to generate operating system metadata for an unmapped virtual storage device (e.g., a VLUN) (block 1510) and make the metadata accessible to a host 110 (block 1515). A first layer of a storage software stack at host 110, such as disk driver layer 114 of FIG. 1b, may be configured to use the O.S. metadata to detect the existence of the virtual storage device as an addressable storage device (e.g., as a LUN) (block 1520), for example during system initialization or boot. After the unmapped virtual storage device is detected, off-host virtualizer 180 may be configured to dynamically map physical storage from one or more back-end physical storage devices 340 (e.g., PLUNs) to an address range within the virtual storage device (block 1525).

[0067] FIG. 16 is a flow diagram illustrating aspects of the operation of a host 110 and an off-host virtualizer 180 according to one embodiment, where off-host virtualizer 180 is configured to support dynamic volume tunneling. The first three blocks depicted in FIG. 16 may represent functionality similar to the first three blocks shown in FIG. 15. That is, off-host virtualizer 180 may be configured to receive operating system metadata requirements (block 1605), generate operating system metadata for an unmapped virtual storage device (e.g., a VLUN) (block 1610) and make the metadata accessible to a host 110 (block 1615). A first layer of a storage software stack, such as disk driver layer 114 of FIG. 1b, may be configured to use the O.S. metadata to detect the existence of the virtual storage device as an addressable storage device (e.g., as a LUN) (block 1620), for example during system initialization or boot. In addition, off-host virtualizer 180 may be configured to aggregate storage at one or physical storage devices 340 into a logical volume (block 1625), and to dynamically map the logical volume to an address range within the previously unmapped virtual storage device (block 1630). Off-host virtualizer 180 may further be configured to provide logical volume metadata to a second layer of the storage software stack at host 110 (e.g., intermediate driver layer 113) (block 1635), allowing the second layer to locate the blocks of the logical volume and to perform desired I/O operations on the logical volume (block 1640).heer

[0068] In various embodiments, off-host virtualizer 180 may implement numerous different types of storage functions using block virtualization. For example, in one embodiment a virtual block device such as a logical volume may implement device striping, where data blocks may be distributed among multiple physical or logical block devices, and/or device spanning, in which multiple physical or logical block devices may be joined to appear as a single large logical block device. In some embodiments, virtualized block devices may provide mirroring and

12/21/2022 19:09:40 Page 19 of 52
SAMSUNG EX1002, Page 293 of 1031

other forms of redundant data storage, the ability to create a snapshot or static image of a particular block device at a point in time, and/or the ability to replicate data blocks among storage systems connected through a network such as a local area network (LAN) or a wide area network (WAN), for example. Additionally, in some embodiments virtualized block devices may implement certain performance optimizations, such as load distribution, and/or various capabilities for online reorganization of virtual device structure, such as online data migration between devices. In other embodiments, one or more block devices may be mapped into a particular virtualized block device, which may be in turn mapped into still another virtualized block device, allowing complex storage functions to be implemented with simple block devices. More than one virtualization feature, such as striping and mirroring, may thus be combined within a single virtual block device in some embodiments, creating a logically hierarchical virtual storage device.

[0069] The off-host virtualizer 180, either alone or in cooperation with one or more other virtualizers such as a volume manager at host 110 or other off-host virtualizers, may provide functions such as configuration management of virtualized block devices and distributed coordination of block device virtualization. For example, after a reconfiguration of a logical volume shared by two hosts 110 (e.g., when the logical volume is expanded, or when a new mirror is added to the logical volume), the off-host virtualizer 180 may be configured to distribute metadata or a volume description indicating the reconfiguration to the two hosts 110. In one embodiment, once the volume description has been provided to the hosts, the storage stacks at the hosts may be configured to interact directly with various storage devices 340 according to the volume description (i.e., to transform logical I/O requests into physical I/O requests using the volume description). Distribution of a virtualized block device as a volume to one or more virtual device clients, such as hosts 110, may be referred to as distributed block virtualization.

[0070] As noted previously, in some embodiments, multiple layers of virtualization may be employed, for example at the host level as well as at an off-host level, such as at a virtualization switch or at a virtualization appliance. In such embodiments, some aspects of virtualization may be visible to a virtual device consumer such as file system layer 112, while other aspects may be implemented transparently by the off-host level. Further, in some multilayer embodiments, the virtualization details of one block device (e.g., one volume) may be fully defined to a virtual device consumer (i.e., without further virtualization at an off-host level), while the virtualization details of another block device (e.g., another volume) may be partially or entirely transparent to the virtual device consumer.

[0071] In some embodiments, a virtualizer, such as off-host virtualizer 180, may be configured to distribute all defined logical volumes to each virtual device client, such as host 110, present within a system. Such embodiments may be referred to as symmetric distributed block virtualization systems. In other embodiments, specific volumes may be distributed only to respective virtual device consumers or hosts, such that at least one volume is not common to two virtual device consumers. Such embodiments may be referred to as asymmetric distributed block virtualization systems.

[0072] It is noted that off-host virtualizer 180 may be any type of device, external to host 110, that is capable of providing the virtualization functionality, including PLUN and volume tunneling, described above. For example, off-host virtualizer 180 may include a virtualization switch, a virtualization appliance, a special additional host dedicated to providing block virtualization, or an embedded system configured to use application specific integrated circuit (ASIC) or field-programmable gate array (FPGA) technology to execute provide block virtualization functionality. FIG. 17 is a block diagram illustrating an embodiment where off-host virtualizer 180 comprises a

12/21/2022 19:09:40 Page 20 of 52
SAMSUNG EX1002, Page 294 of 1031

virtualization switch 1710, and FIG. 18 is a block diagram illustrating another embodiment where off-host virtualizer 180 comprises a virtualization appliance 1810. A virtualization switch 1710 may be an intelligent fibre channel switch, configured with sufficient processing capacity to perform virtualization functions in addition to providing fibre channel connectivity. A virtualization appliance 1810 may be an intelligent device programmed to provide virtualization functions, such as mirroring, striping, snapshots, and the like. An appliance may differ from a general-purpose computer in that the appliance software is normally customized for the function (such as virtualization) to which the appliance is dedicated, pre-installed by the vendor, and not easily modifiable by a user. In some embodiments, off-host block virtualization may be provided by a collection of cooperating devices, such as two or more virtualizing switches, instead of a single device. Such a collection of cooperating devices may be configured for failover, i.e., a standby cooperating device may be configured to take over the virtualization functions supported by a failed cooperating device. An off-host virtualizer 180 may incorporate one or more processors, as well as volatile and/or non-volatile memory. In some embodiments, configuration information associated with virtualization may be maintained at a database separate from the off-host virtualizer 180, and may be accessed by off-host virtualizer over a network. In one embodiment, an off-host virtualizer may be programmable and/or configurable. Numerous other configurations of off-host virtualizer 180 are possible and contemplated.

[0073] FIG. 19 is a block diagram illustrating an exemplary host 110, according to one embodiment. Host 110 may be any computer system, such as a server comprising one or more processors 1910 and one or more memories 1920, capable of supporting the storage software stack 140B described above. As shown, host 110 may include one or more local storage devices 1930 (such as disks) as well as one or more network interfaces 1940 providing an interface to an interconnect 130. Portions of the storage software stack 140B may be resident in a memory 1920, and may be loaded into memory 1920 from storage devices 1930 as needed. In some embodiments, a host 110 may also be a diskless computer, configured to access storage from a remote location instead of using local storage devices 1930. Various other components, such as video cards, monitors, mice, etc. may also be included within a host 110 in different embodiments. Any desired operating system may be used at a host 110, including various versions of Microsoft Windows™, Solaris™ from Sun Microsystems, various versions of Linux, other operating systems based on UNIX, and the like. The intermediate driver layer 113 may be included within a volume manager in some embodiments.

[0074] FIG. 20 is a block diagram illustrating a computer-accessible medium 2000 comprising virtualization software 2010 capable of providing the functionality of off-host virtualizer 180 and block storage software stack 140B described above. Virtualization software 2010 may be provided to a computer system using a variety of computer-accessible media including electronic media (e.g., flash memory), magnetic media such as RAM (e.g., SDRAM, RDRAM, SRAM, etc.), optical storage media such as CD-ROM, etc., as well as transmission media or signals such as electrical, electromagnetic or digital signals, conveyed via a communication medium such as a network and/or a wireless link.

[0075] Although the embodiments above have been described in considerable detail, numerous variations and modifications will become apparent to those skilled in the art once the above disclosure is fully appreciated. It is intended that the following claims be interpreted to embrace all such variations and modifications.

12/21/2022 19:09:40 Page 21 of 52
SAMSUNG EX1002, Page 295 of 1031

WHAT IS CLAIMED IS:

1. A system comprising:

a first host; and

an off-host virtualizer;

wherein the off-host virtualizer is configured to:

generate first operating system metadata for a first virtual storage device; and

make the first operating system metadata accessible to the first host;

wherein the first host comprises a storage software stack including a first layer;

wherein the first layer is configured to use the first operating system metadata to detect an existence of the first virtual storage device as a first addressable storage device.

2. The system as recited in Claim 1, wherein the off-host virtualizer is further configured to customize the operating system metadata in response to a requirement of a particular operating system in use at the first host.

3. The system as recited in Claim 2, wherein the first host is configured to provide the requirement of the particular operating system to the off-host virtualizer.

4. The system as recited in Claim 1, further comprising a storage device, wherein the first addressable storage device is a partition of the storage device.

5. The system as recited in Claim 1, further comprising one or more physical storage devices, wherein the storage software stack includes a second layer, wherein the off-host virtualizer is further configured to:

aggregate storage in the one or more physical storage devices into a logical volume;

map the logical volume to an address space of the first virtual storage device;

provide logical volume metadata for the logical volume to the second layer of the storage software stack;

wherein the second layer of the storage software stack is configured to use the logical volume metadata to access the logical volume for I/O operations.

6. The system as recited in Claim 1, wherein the first virtual storage device is a virtual logical unit (LUN).

7. The system as recited in Claim 1, wherein the off-host virtualizer includes a virtualization switch.

8. The system as recited in Claim 1, wherein the off-host virtualizer includes a virtualization appliance.

9. The system as recited in Claim 1, wherein the first virtual storage device is unmapped to physical storage.

10. The system as recited in Claim 9, further comprising two or more physical storage devices, wherein the off-host virtualizer is further configured to dynamically map each physical storage device of the two or more physical storage devices to a respective address range within the first virtual storage device.

12/21/2022 19:09:40 Page 22 of 52
SAMSUNG EX1002, Page 296 of 1031

11. The system as recited in recited in Claim 9, further comprising a physical storage device and a second host, wherein the off-host virtualizer is further configured to:

generate second operating system metadata for a second virtual storage device, wherein the second virtual storage device is unmapped to physical storage; and

make the second operating system metadata accessible to the second host;

wherein the second host comprises a second storage software stack including a first layer;

wherein the first layer of the second storage software stack is configured to use the second operating system metadata to detect an existence of the second virtual storage device as a second addressable storage device;

wherein the off-host virtualizer is further configured to:

dynamically map a first physical address range of the physical storage device to a first virtual address range within the first virtual storage device;

dynamically map a second physical address range of the physical storage device to a second virtual address range within the second virtual storage device; and

prevent access to the second physical address range from the first host.

12. The system as recited in Claim 9, further comprising a first physical storage device, wherein the storage software stack includes a second layer, wherein the off-host virtualizer is further configured to:

aggregate a first range of physical storage within the first physical storage device into a first logical volume;

dynamically map a first range of virtual storage within the first virtual storage device to the first logical volume; and

provide first logical volume metadata for the first logical volume to the second layer of the storage software stack;

wherein the second layer of the storage software stack is configured to use the first logical volume metadata to access the first logical volume for I/O operations.

13. The system as recited in Claim 12, wherein the off-host virtualizer is further configured to:

aggregate a second range of physical storage within the first physical storage device into a second logical volume;

dynamically map a second range of virtual storage within the first virtual storage device to the second logical volume; and

provide additional logical volume metadata for the second logical volume to the second layer of the storage software stack;

wherein the second layer of the storage software stack is configured to use the additional logical volume metadata to access the second logical volume for I/O operations.

14. The system as recited in Claim 12, further comprising a second host, wherein the off-host virtualizer is further configured to:

12/21/2022 19:09:40 Page 23 of 52
SAMSUNG EX1002, Page 297 of 1031

generate second operating system metadata for a second virtual storage device, wherein the second virtual storage device is unmapped to physical storage;

make the second operating system metadata accessible to the second host;

aggregate a second range of physical storage within the first physical storage device into a second logical volume; and

provide second logical volume metadata for the second logical volume to the second host;

wherein the second host comprises a second storage software stack comprising a first and a second layer, wherein the first layer of the second software stack is configured to use the second operating system metadata to detect an existence of the second virtual storage device as a second addressable storage device, and wherein the second layer of the second software stack is configured to use the second logical volume metadata to access the second logical volume for I/O operations.

15. The system as recited in Claim 14, wherein the off-host virtualizer is further configured to prevent the second host from performing an I/O operation on the first logical volume.

16. The system as recited in Claim 14, wherein the first addressable storage device is accessible to the first host via a first storage network, and the second addressable storage device is accessible to the second host via a second storage network.

17. The system as recited in Claim 12, further comprising a second host and a second physical storage device, wherein the off-host virtualizer is further configured to:

generate second operating system metadata for a second virtual storage device, wherein the second virtual storage device is unmapped to physical storage;

make the second operating system metadata accessible to the second host;

aggregate a second range of physical storage within the second physical storage device into a second logical volume;

dynamically map a second range of virtual storage within the first virtual storage device to the second logical volume; and

provide second logical volume metadata for the second logical volume to the second host;

wherein the second host comprises a second storage software stack comprising a first and a second layer, wherein the first layer of the second software stack is configured to use the second operating system metadata to detect an existence of the second virtual storage device as a second addressable storage device, and wherein the second layer of the second software stack is configured to use the second logical volume metadata to access the second logical volume for I/O operations;

wherein the first addressable storage device is accessible to the first host via a first storage network, the second addressable storage device is accessible to the second host via a second storage network, the first physical storage device is accessible to the off-host virtualizer via a third storage network, and the second physical storage device is accessible to the off-host virtualizer via a fourth storage network.

12/21/2022 19:09:40 Page 24 of 52
SAMSUNG EX1002, Page 298 of 1031

18. The system as recited in Claim 17, wherein each storage network of the first, second, third and fourth storage networks includes an independently configurable fibre channel fabric.

19. The system as recited in Claim 1, wherein the storage software stack includes a second layer, further comprising a first and a second physical storage device, wherein the off-host virtualizer is further configured to:

aggregate a first range of storage within the first physical storage device and a second range of storage within the second physical storage device into a single logical volume; and

provide logical volume metadata for the single logical volume to the second layer of the storage software stack;

wherein the second layer of the storage software stack is configured to use the logical volume metadata to access the single logical volume for I/O operations.

20. A virtualizer operable to generate operating system metadata for a storage device, wherein the operating system metadata emulates a storage volume hosted under a particular operating system.

21. A method comprising:

generating first operating system metadata for a first virtual storage device;

making the first operating system metadata for the first virtual storage device accessible to a first host; and

using the first operating system metadata at a first layer of a storage software stack at the first host to detect an existence of the first virtual storage device as a first addressable storage device.

22. The method as recited in Claim 21, further comprising:

customizing the operating system metadata in response to a requirement of a particular operating system in use at the first host.

23. The method as recited in Claim 21, wherein the first virtual storage device is unmapped to physical storage.

24. The method as recited in Claim 23, further comprising:

dynamically mapping a first and a second physical storage device to a respective address range within the first virtual storage device.

25. The method as recited in Claim 23, further comprising:

aggregating a first range of physical storage within a first physical storage device into a first logical volume;

dynamically mapping a first range of virtual storage within the first virtual storage device to the first logical volume;

providing first logical volume metadata for the first logical volume to a second layer of the storage software stack; and

using the first logical volume metadata to access the first logical volume for I/O operations from the second layer of the storage software stack.

12/21/2022 19:09:40 Page 25 of 52
SAMSUNG EX1002, Page 299 of 1031

26. The method as recited in Claim 25, further comprising:
aggregating a second range of physical storage within the first physical storage device into a second logical volume;
dynamically mapping a second range of virtual storage within the first virtual storage device to the second logical volume;
providing additional logical volume metadata for the second logical volume to the second layer of the storage software stack; and
using the additional logical volume metadata at the second layer of the storage software stack to access the second logical volume for I/O operations.

27. The method as recited in Claim 25, further comprising:
generating second operating system metadata for a second virtual storage device, wherein the second virtual storage device is unmapped to physical storage;
making the second operating system metadata accessible to a second host;
using the second operating system metadata at a first layer of a storage software stack at the second host to detect an existence of the second virtual storage device as a second addressable storage device.
aggregating a second range of physical storage within the first physical storage device into a second logical volume;
dynamically mapping a second range of virtual storage within the second virtual storage device to the second logical volume;
providing second logical volume metadata for the second logical volume to a second layer of the storage software stack at the second host; and
using the second logical volume metadata to access the second logical volume for I/O operations from the second layer of the storage software stack at the second host.

28. The method as recited in Claim 27, further comprising:
preventing the second host from performing an I/O operation on the first logical volume.

29. The method as recited in Claim 25, further comprising:
generating second operating system metadata for a second virtual storage device, wherein the second virtual storage device is unmapped to physical storage;
making the second operating system metadata accessible to a second host;
using the second operating system metadata at a first layer of a storage software stack at the second host to detect an existence of the second virtual storage device as a second addressable storage device.
aggregating a second range of physical storage within a second physical storage device into a second logical volume at an off-host virtualizer;
dynamically mapping a second range of virtual storage within the second virtual storage device to the second logical volume;

12/21/2022 19:09:40 Page 26 of 52
SAMSUNG EX1002, Page 300 of 1031

providing second logical volume metadata for the second logical volume to a second layer of the storage software stack at the second host; and

using the second logical volume metadata to access the second logical volume for I/O operations from the second layer of the storage software stack at the second host;

wherein the first range of physical storage is aggregated into the first logical volume by the off host virtualizer; and

wherein the first addressable storage device is accessible to the first host via a first storage network, the second addressable storage device is accessible to the second host via a second storage network, the first physical storage device is accessible to an off-host virtualizer via a third storage network, and the second physical storage device is accessible to the off-host virtualizer via a fourth storage network.

30. The method as recited in Claim 29, wherein each storage network of the first, second, third and fourth storage networks includes an independently configurable fibre channel fabric.

31. The method as recited in Claim 23, further comprising:

aggregating a first range of storage within a first physical storage device and a second range of storage within a second physical storage device into a single logical volume;

dynamically mapping a range of virtual storage within the virtual storage device to the second logical volume;

providing logical volume metadata for the single logical volume to a second layer of the storage software stack; and

using the logical volume metadata to access the single logical volume for I/O operations from the second layer of the storage software stack.

32. A computer accessible medium comprising program instructions, wherein the instructions are executable to:

generate first operating system metadata for a first virtual storage device;

make the first operating system metadata for the first virtual storage device accessible to a first host; and

use the first operating system metadata at a first layer of a storage software stack at the host to detect an existence of the first virtual storage device as a first addressable storage device.

33. The computer accessible medium as recited in Claim 32, wherein the instructions are further executable to:

customize the operating system metadata in response to a requirement of a particular operating system in use at the first host.

34. The computer accessible medium as recited in Claim 32, wherein the first virtual storage device is unmapped to physical storage.

35. The computer accessible medium as recited in Claim 34, wherein the instructions are further executable to:

map a first and a second physical storage device to a respective address range within the first virtual storage device.

12/21/2022 19:09:40 Page 27 of 52
SAMSUNG EX1002, Page 301 of 1031

36. The computer accessible medium as recited in Claim 34, wherein the instructions are further executable to:

aggregate a first range of physical storage within a first physical storage device into a first logical volume;

dynamically map a first range of virtual storage within the first virtual storage device to the first logical volume;

provide first logical volume metadata for the first logical volume to a second layer of the storage software stack; and

use the first logical volume metadata to access the first logical volume for I/O operations from the second layer of the storage software stack.

12/21/2022 19:09:40 Page 28 of 52
SAMSUNG EX1002, Page 302 of 1031

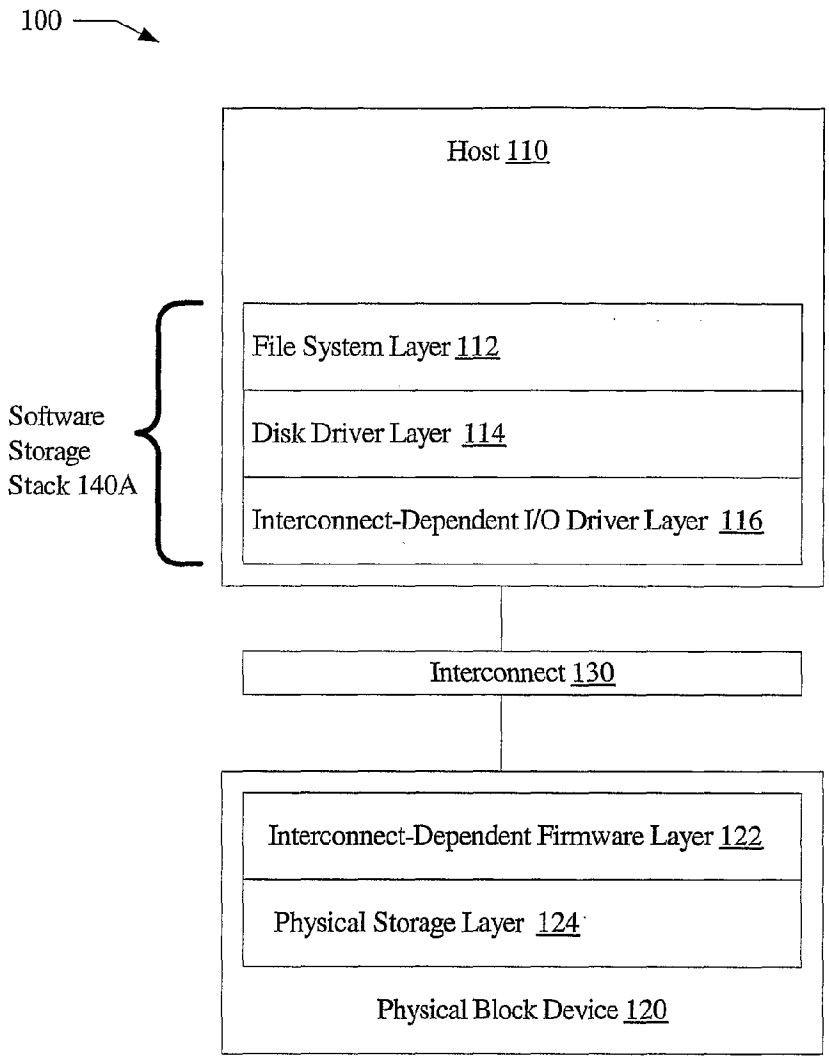


FIG. 1a

SAMSUNG EX1002, Page 303 of 1031

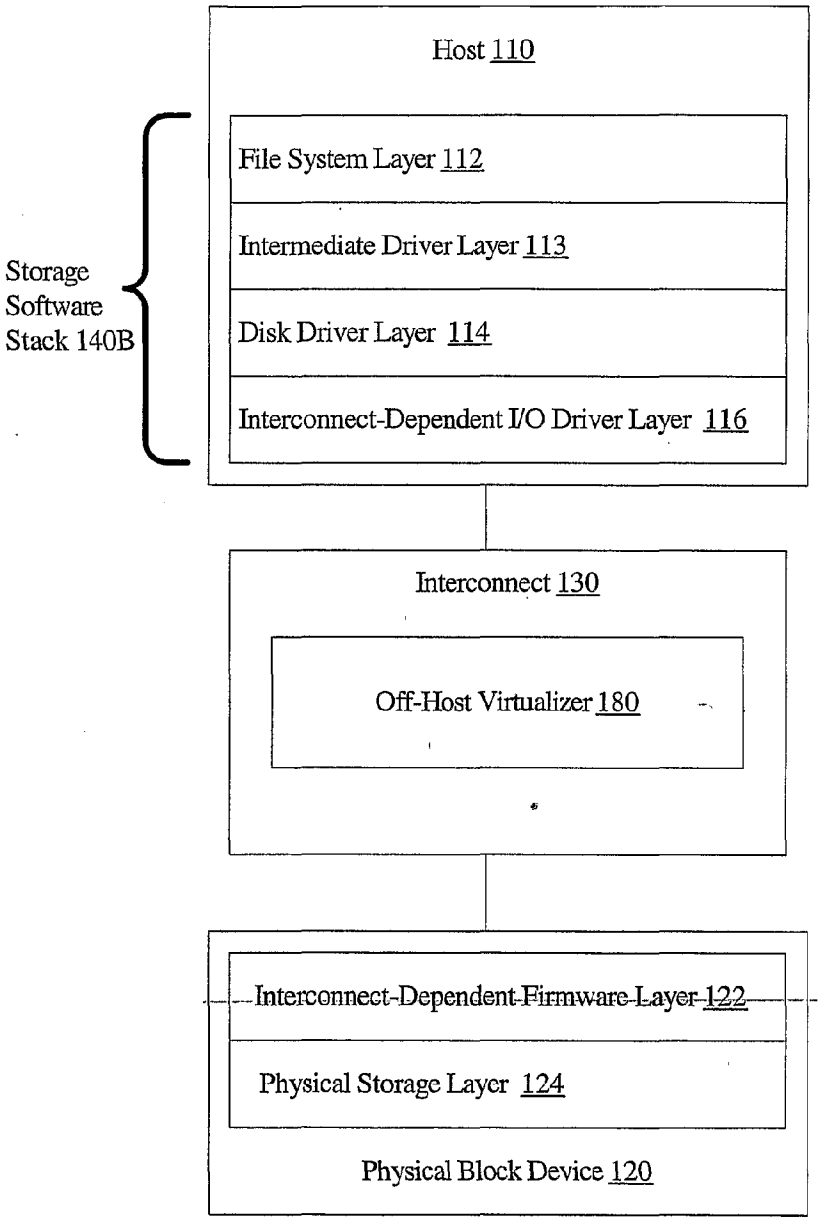


FIG. 1b

SAMSUNG EX1002, Page 304 of 1031

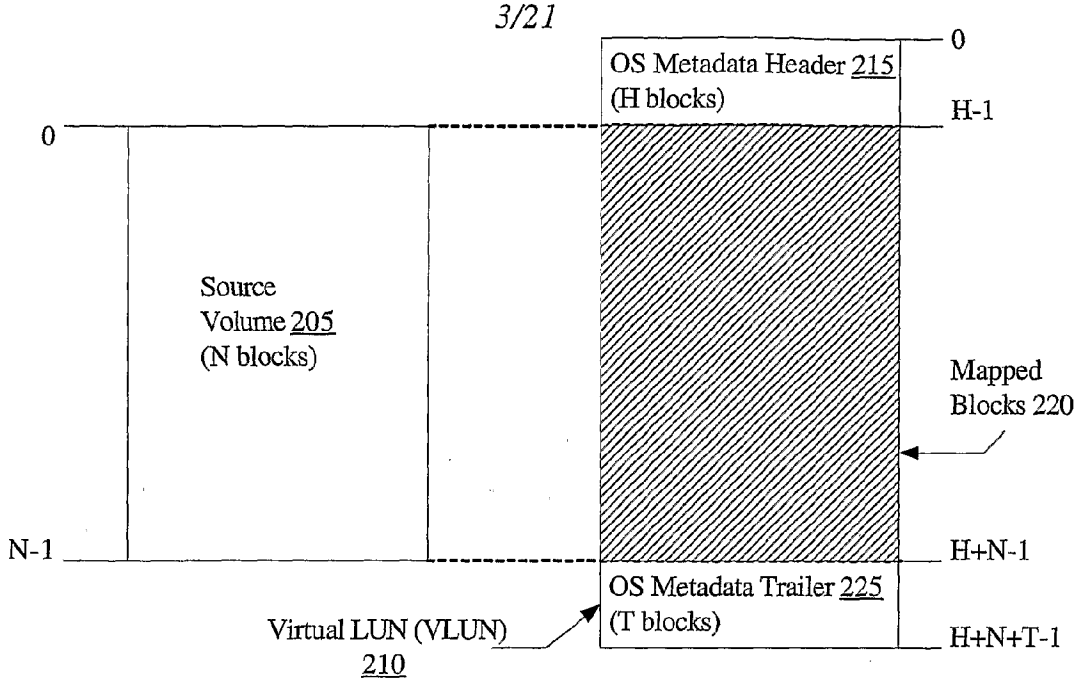


FIG. 2a

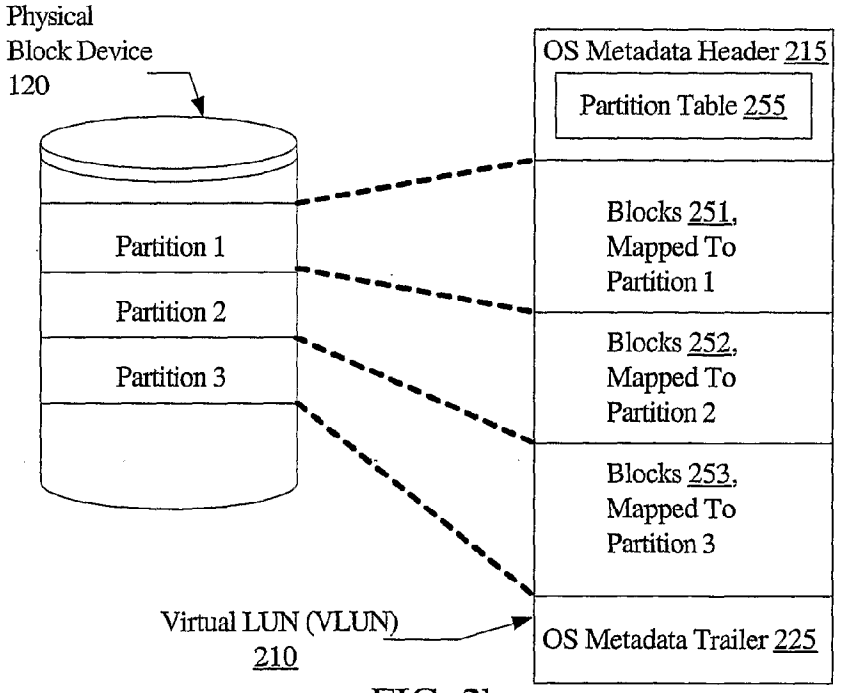


FIG. 2b

SAMSUNG EX1002, Page 305 of 1031

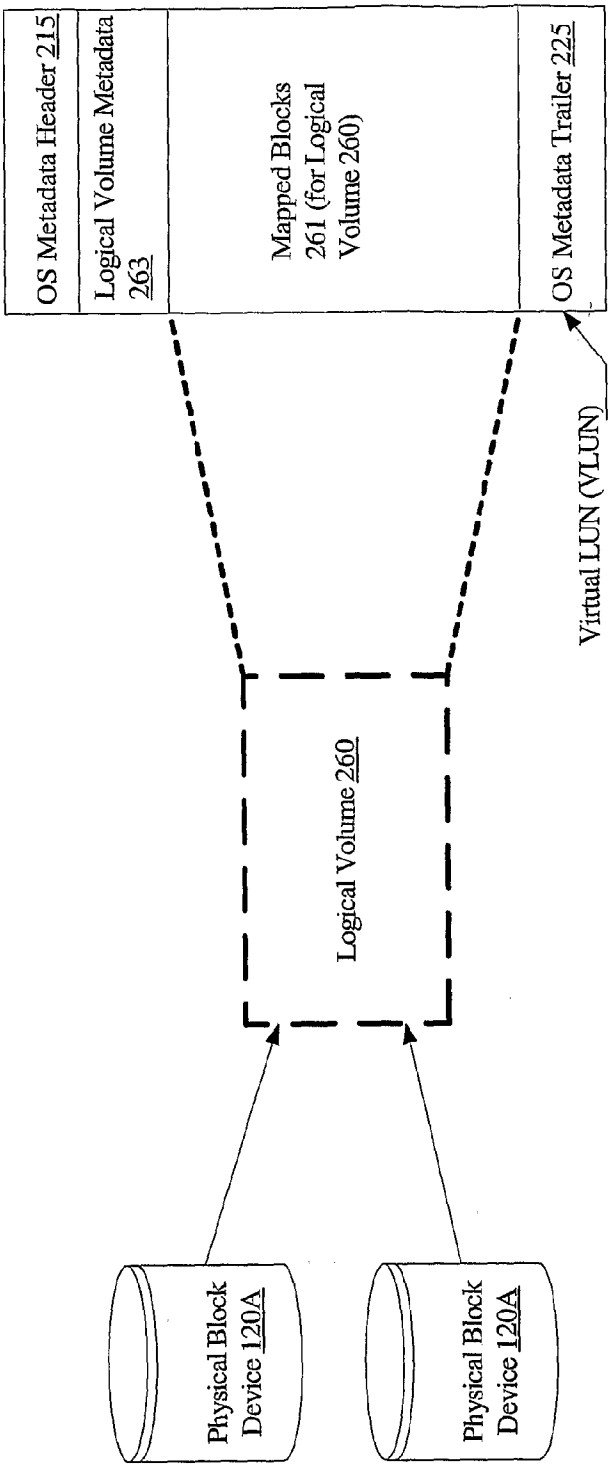


FIG. 2c

SAMSUNG EX1002, Page 306 of 1031

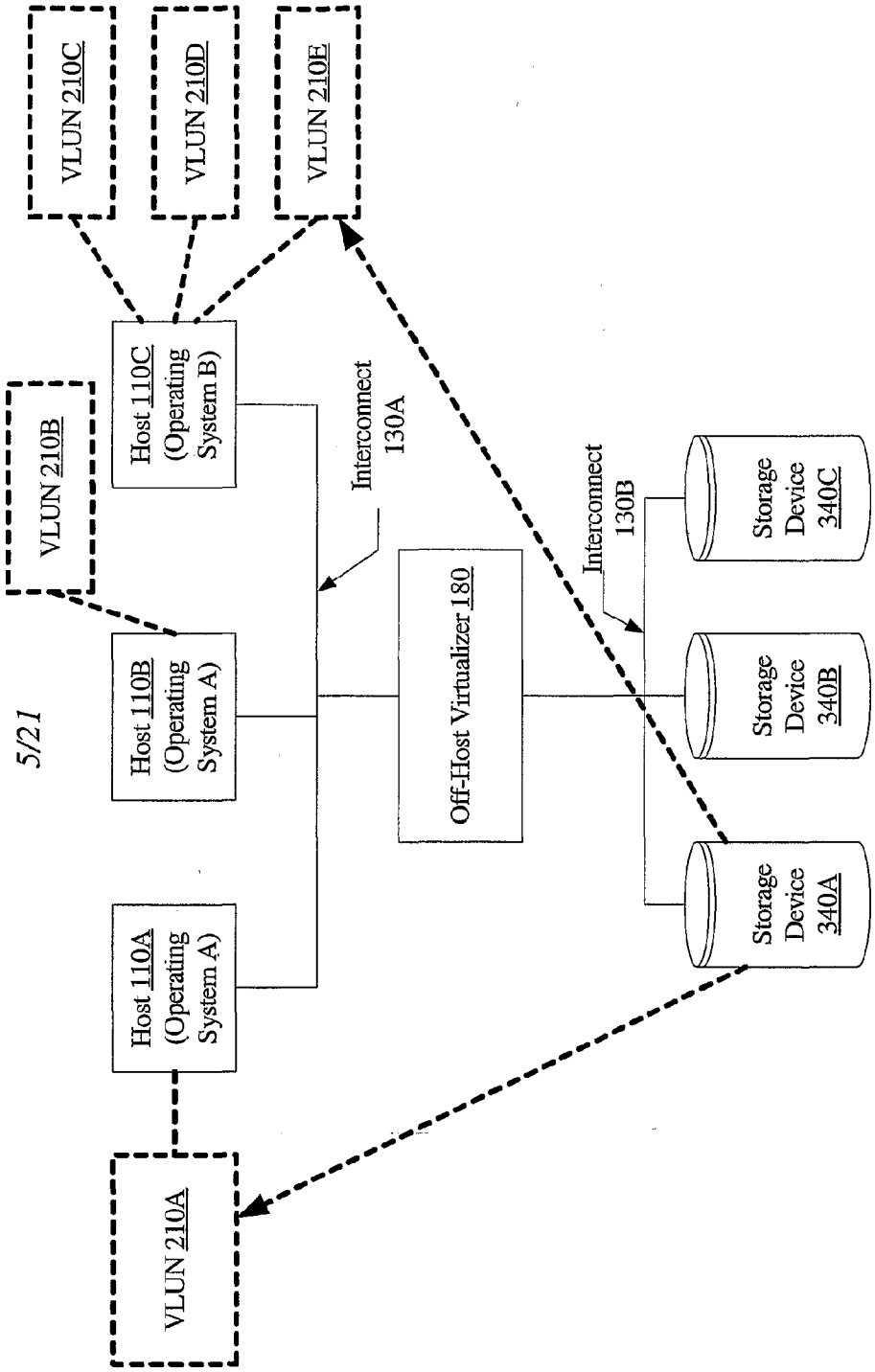


FIG. 3

SAMSUNG EX1002, Page 307 of 1031

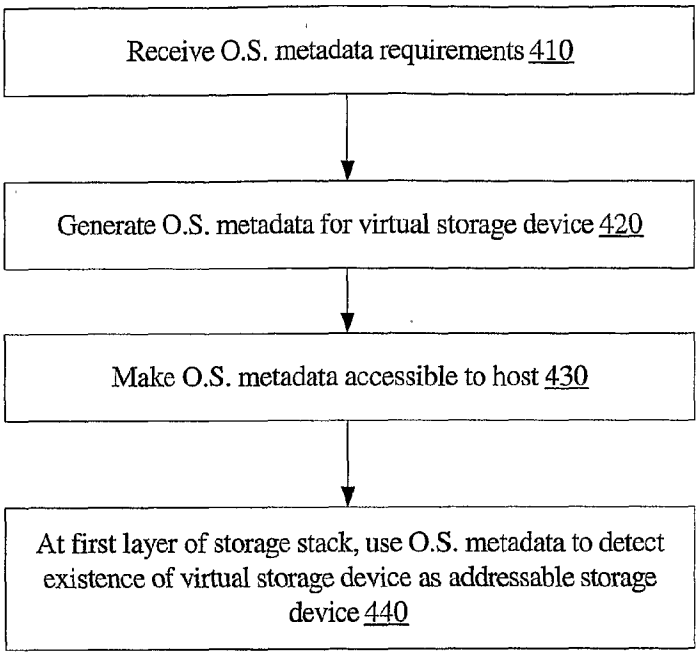


FIG. 4

SAMSUNG EX1002, Page 308 of 1031

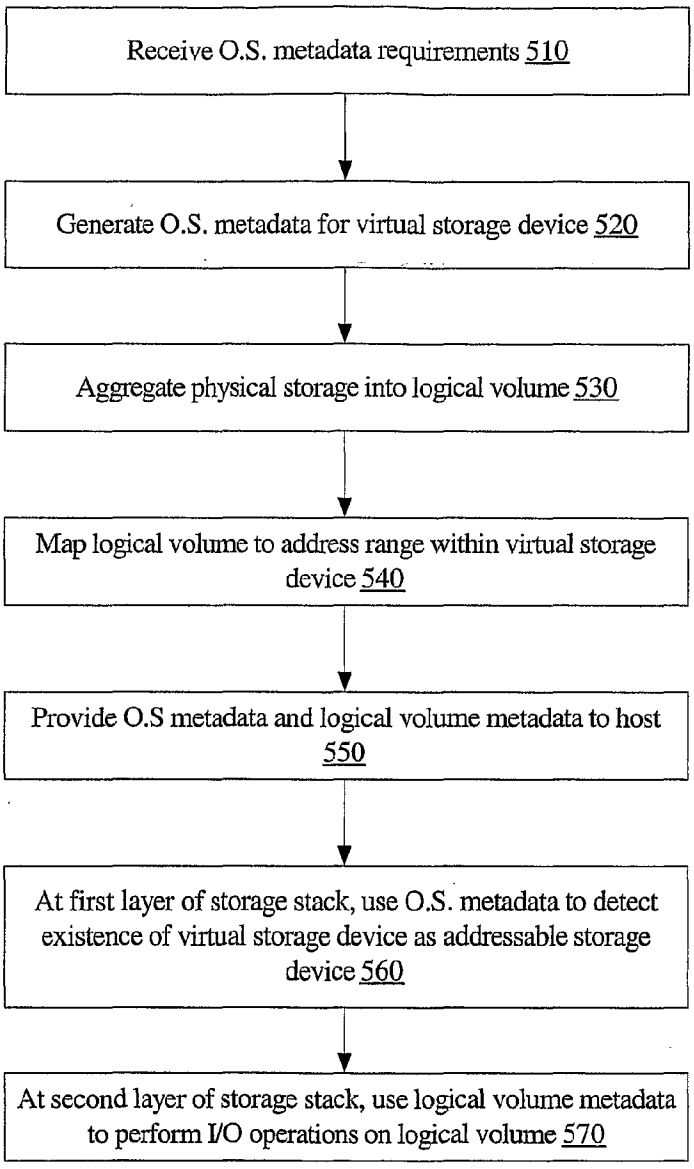


FIG. 5

SAMSUNG EX1002, Page 309 of 1031

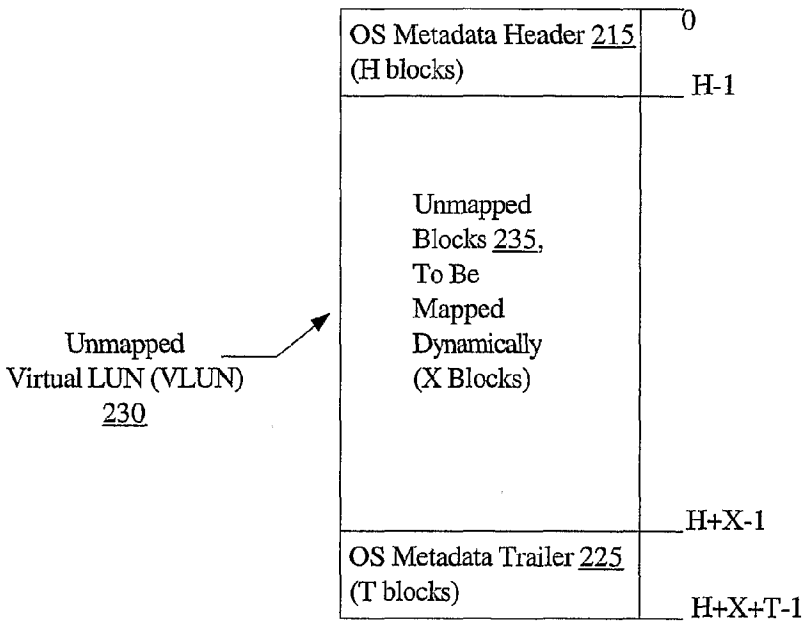


FIG. 6

SAMSUNG EX1002, Page 310 of 1031

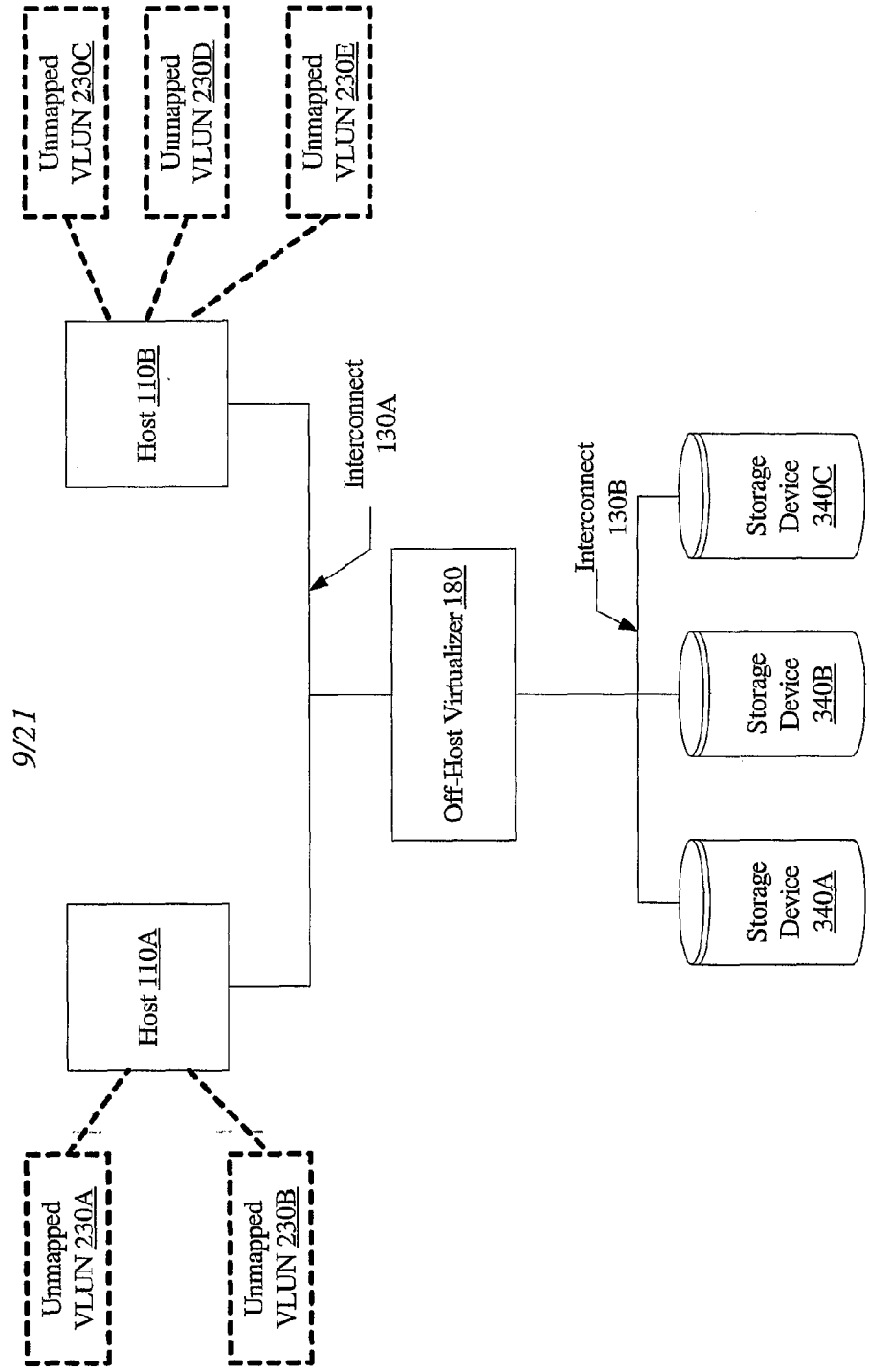


FIG. 7

SAMSUNG EX1002, Page 311 of 1031

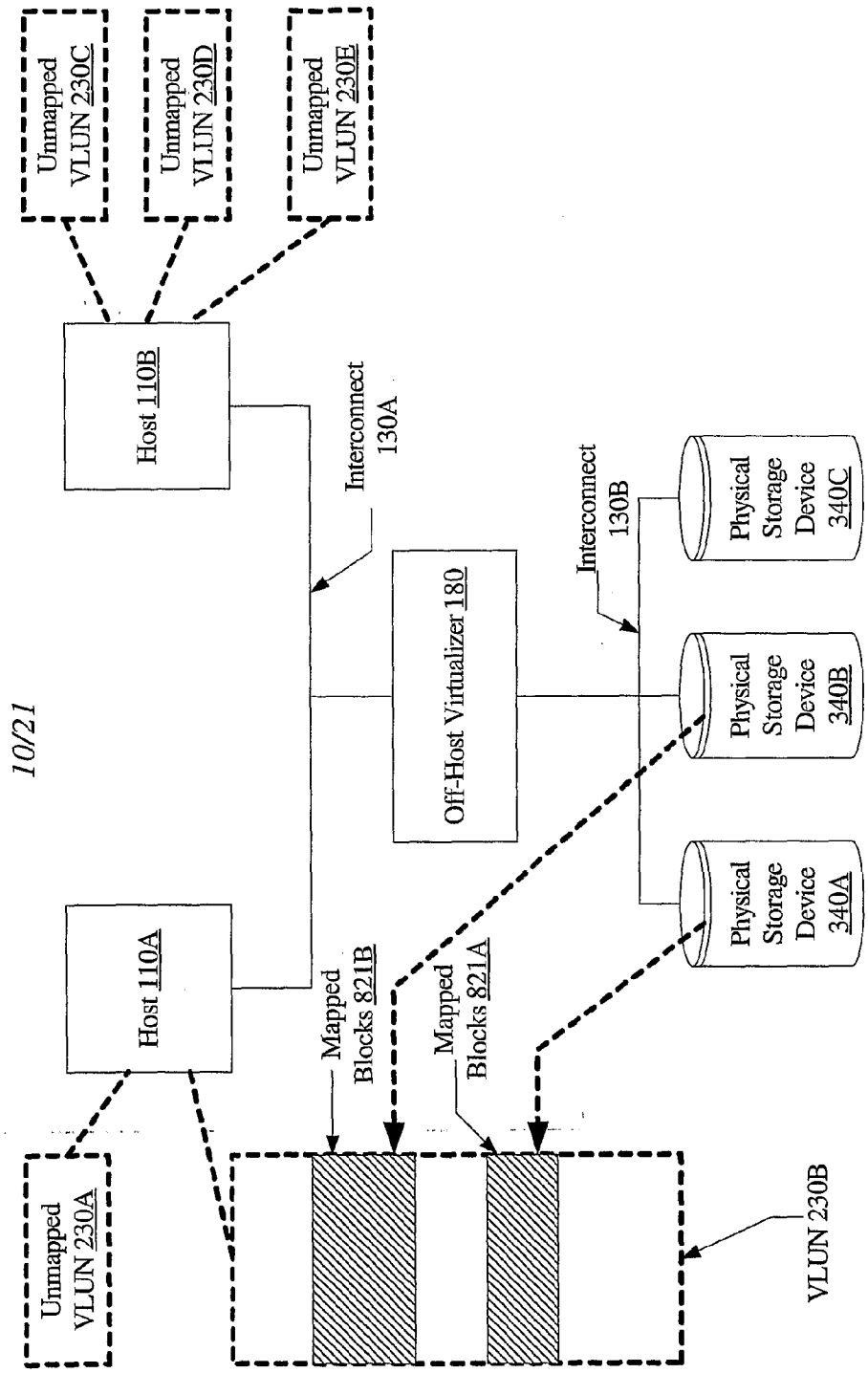

Unmapped VLUN 230C
Unmapped VLUN 230D
Unmapped VLUN 230E
Host 110B
Interconnect 130A
Interconnect 130B
Off-Host Virtualizer 180
Physical Storage Device 340C
Physical Storage Device 340B
Physical Storage Device 340A
Host 110A
Mapped Blocks 821B
Mapped Blocks 821A
Unmapped VLUN 230A
VLUN 230B

FIG. 8

10/21

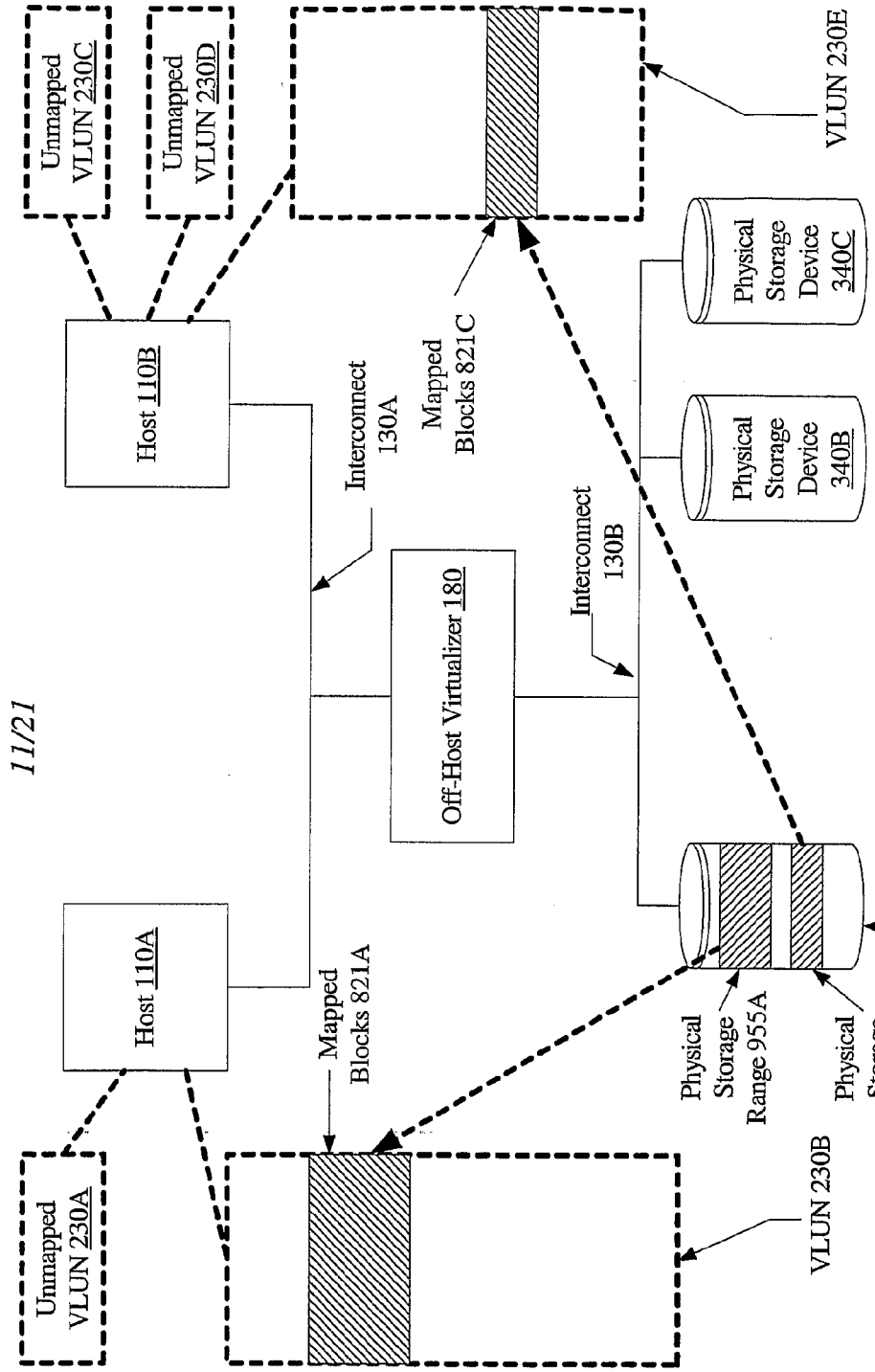

Unmapped VLUN 230C
Unmapped VLUN 230D
VLUN 230E
Host 110B
Interconnect 130A
Mapped Blocks 821C
Physical Storage Device 340C
Physical Storage Device 340B
Interconnect 130B
Off-Host Virtualizer 180
Host 110A
Mapped Blocks 821A
Physical Storage Device 340A
Physical Storage Range 955A
Physical Storage Range 955B
Unmapped VLUN 230A
VLUN 230B

FIG. 9

11/21

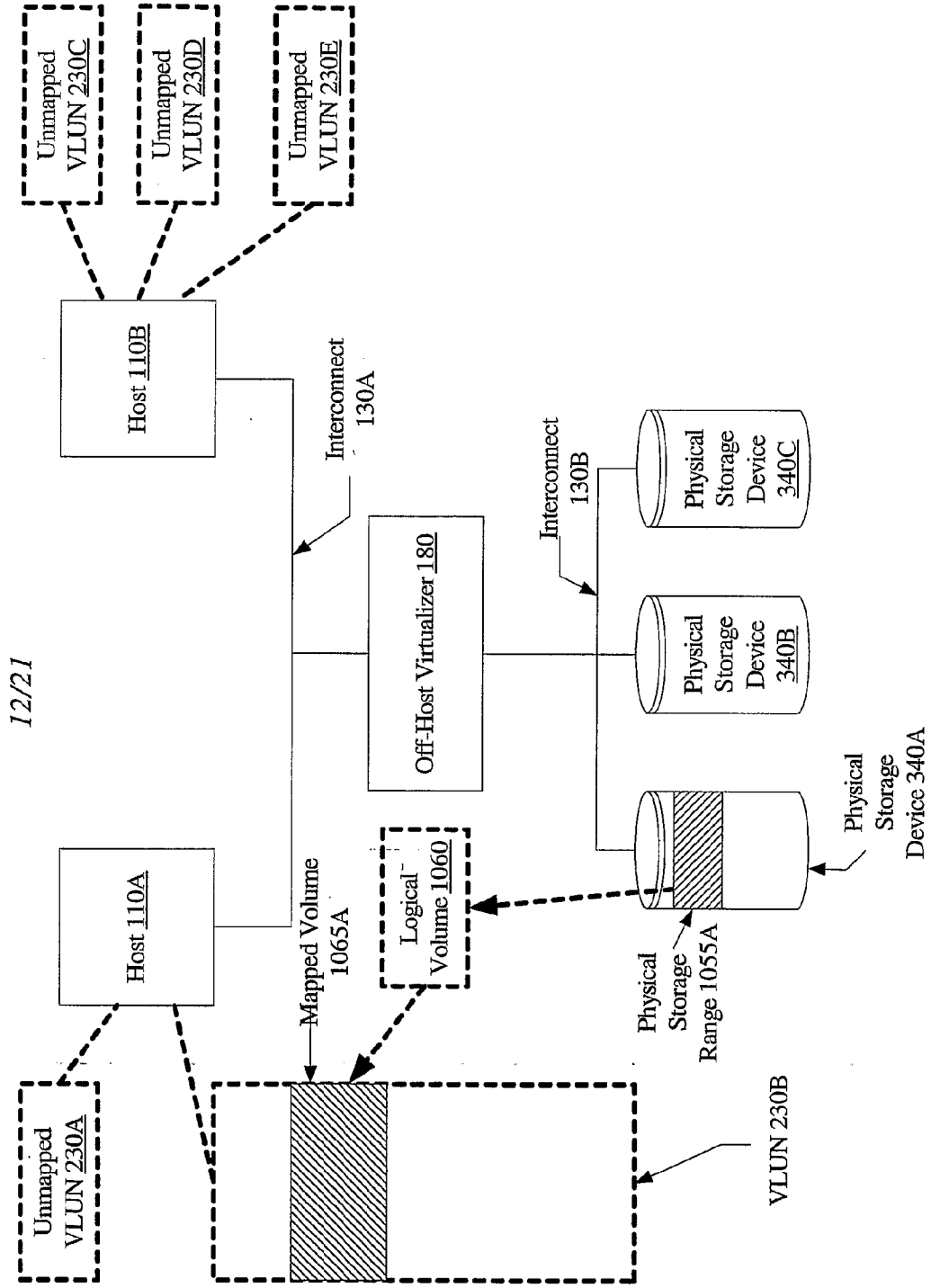

Unmapped VLUN 230C
Unmapped VLUN 230D
Unmapped VLUN 230E
Host 110B
Interconnect 130A
Interconnect 130B
Physical Storage Device 340C
Off-Host Virtualizer 180
Physical Storage Device 340B
Physical Storage Device 340A
Host 110A
Logical Volume 1060
Mapped Volume 1065A
Physical Storage Range 1055A
Unmapped VLUN 230A
VLUN 230B

FIG. 10

12/21

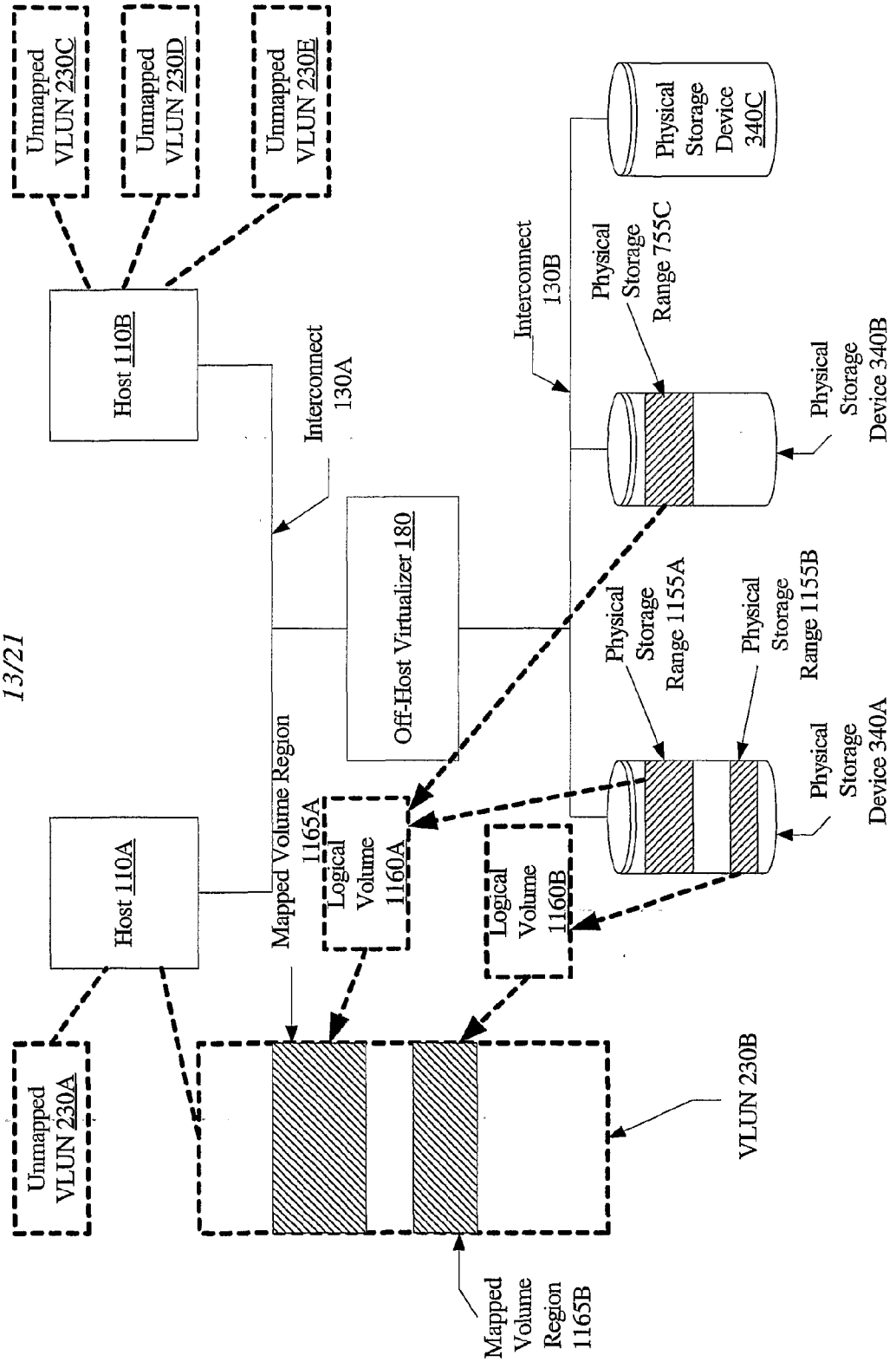

Unmapped VLUN 230C
Unmapped VLUN 230D
Unmapped VLUN 230E
Host 110B
Interconnect 130A
Interconnect 130B
Physical Storage Device 340C
Physical Storage Range 755C
Physical Storage Device 340B
Off-Host Virtualizer 180
Physical Storage Range 1155A
Physical Storage Range 1155B
Physical Storage Device 340A
Host 110A
Mapped Volume Region 1165A
Logical Volume 1160A
Logical Volume 1160B
Unmapped VLUN 230A
VLUN 230B
Mapped Volume Region 1165B

FIG. 11

13/21

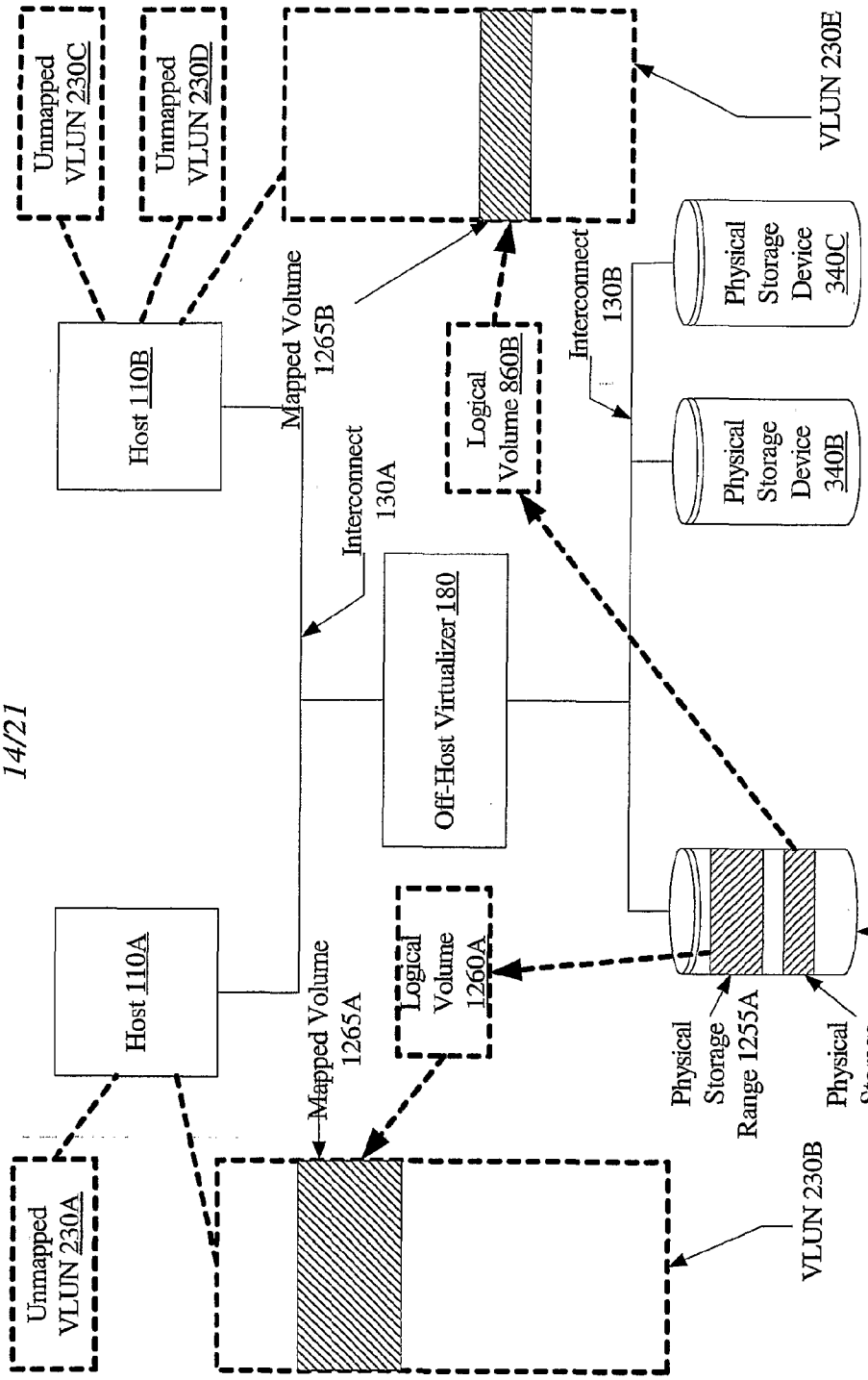

Unmapped VLUN 230C
Unmapped VLUN 230D
VLUN 230E
Host 110B
Mapped Volume 1265B
Interconnect 130A
Logical Volume 860B
Interconnect 130B
Physical Storage Device 340C
Physical Storage Device 340B
Off-Host Virtualizer 180
Host 110A
Mapped Volume 1265A
Logical Volume 1260A
Physical Storage Device 340A
Physical Storage Range 1255A
Physical Storage Range 1255B
Unmapped VLUN 230A
VLUN 230B

FIG. 12

14/21

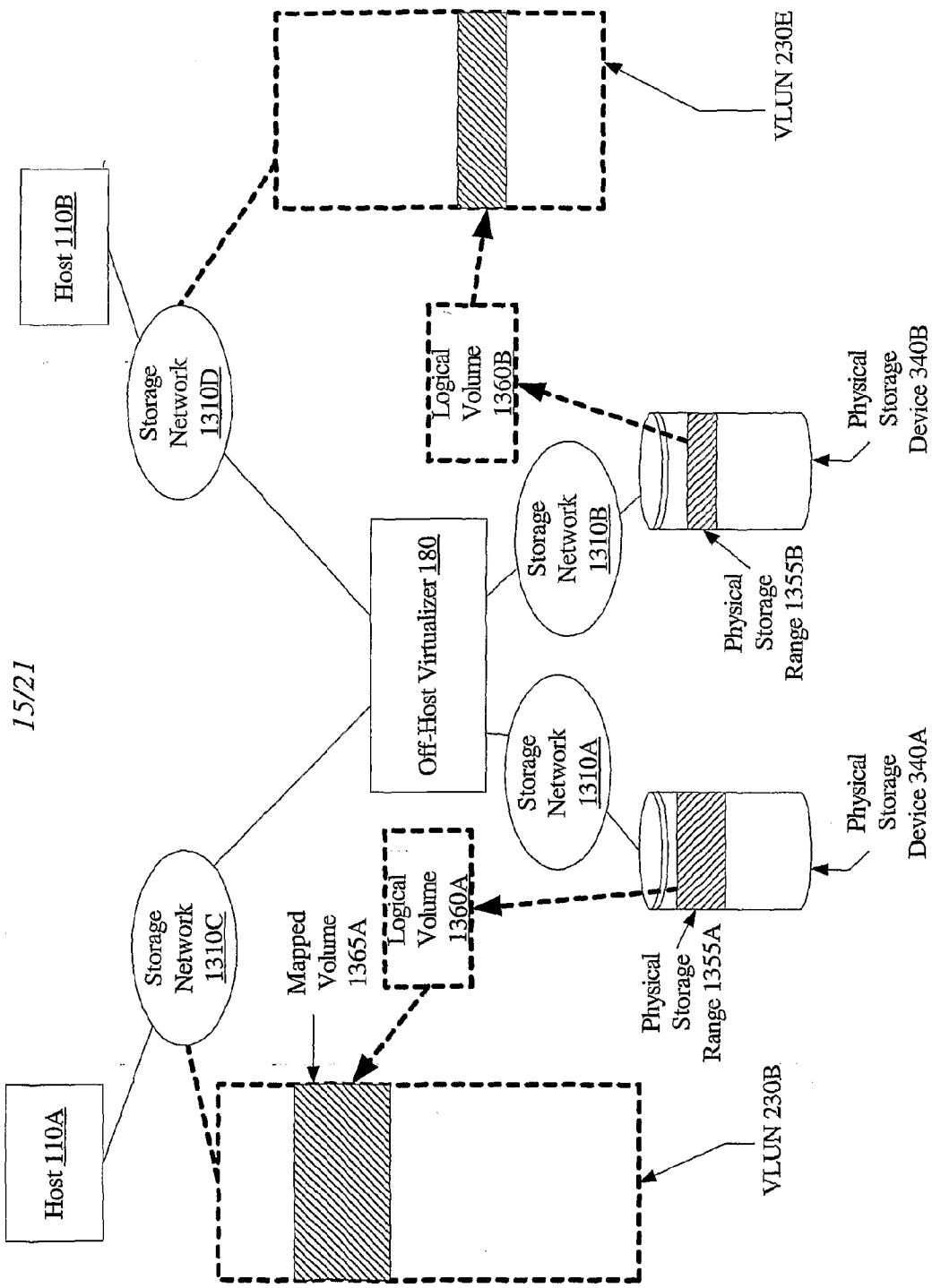

Host 110B
Host 110A
Storage Network 1310D
Storage Network 1310C
Storage Network 1310B
Storage Network 1310A
Off-Host Virtualizer 180
Logical Volume 1360B
Logical Volume 1360A
Mapped Volume 1365A
VLUN 230E
VLUN 230B
Physical Storage Device 340B
Physical Storage Device 340A
Physical Storage Range 1355B
Physical Storage Range 1355A

FIG. 13

15/21

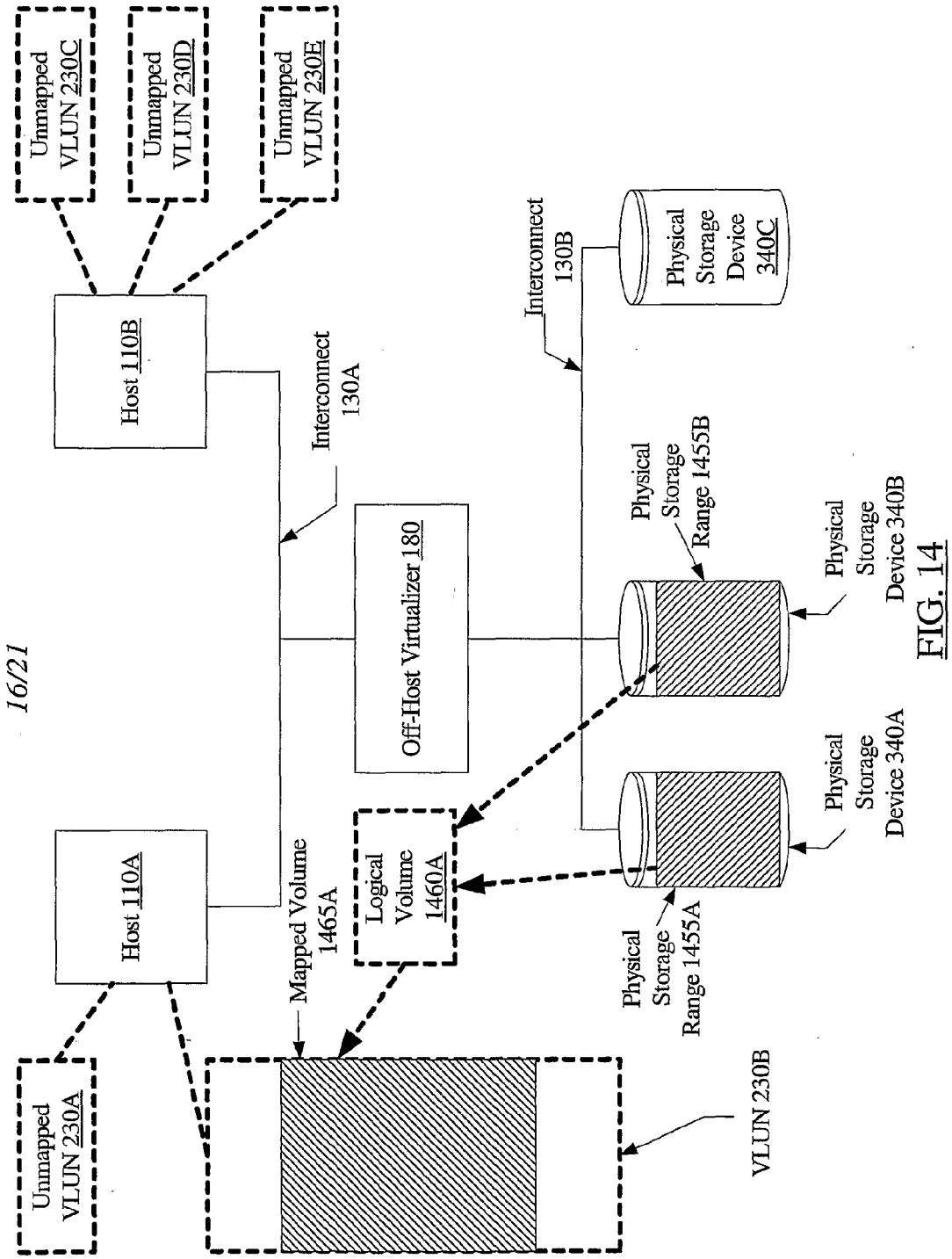


FIG. 14

16/21

SAMSUNG EX1002, Page 318 of 1031

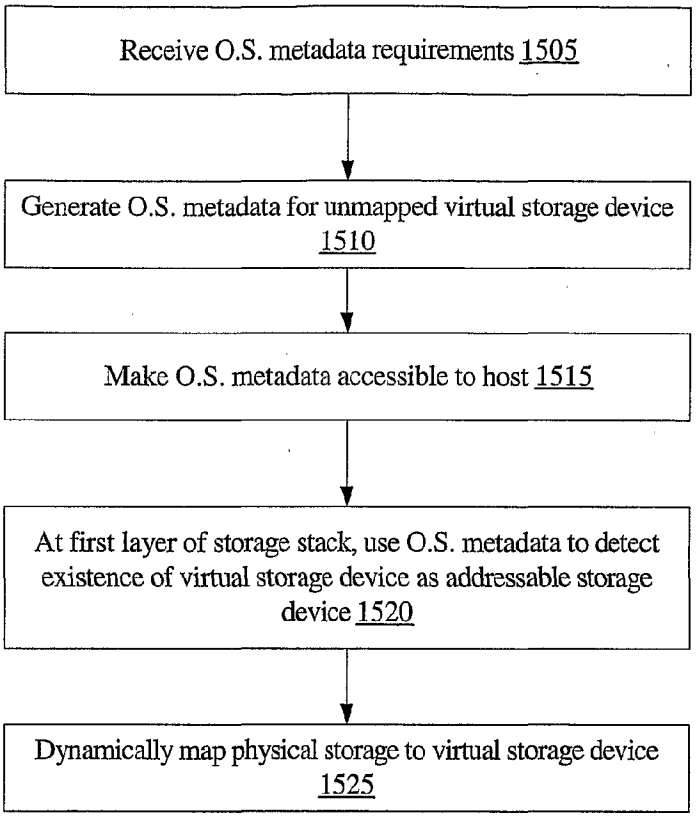


FIG. 15

SAMSUNG EX1002, Page 319 of 1031

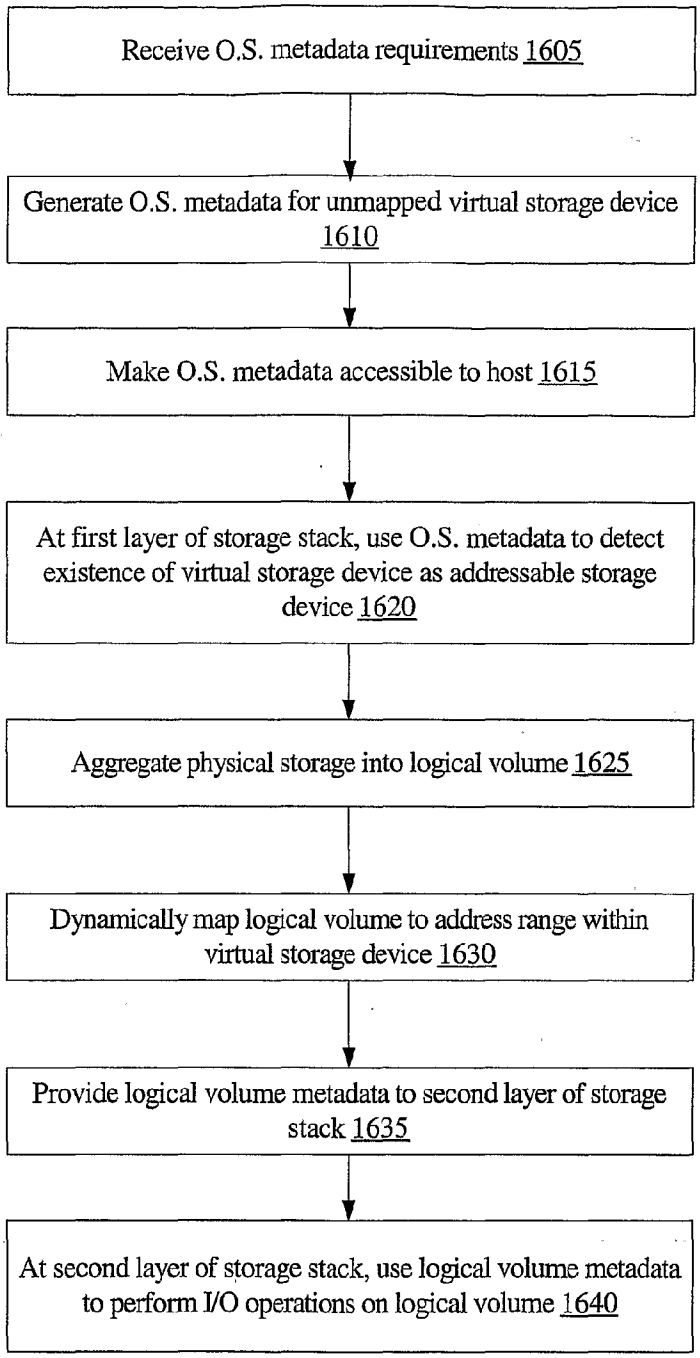


FIG. 16

SAMSUNG EX1002, Page 320 of 1031

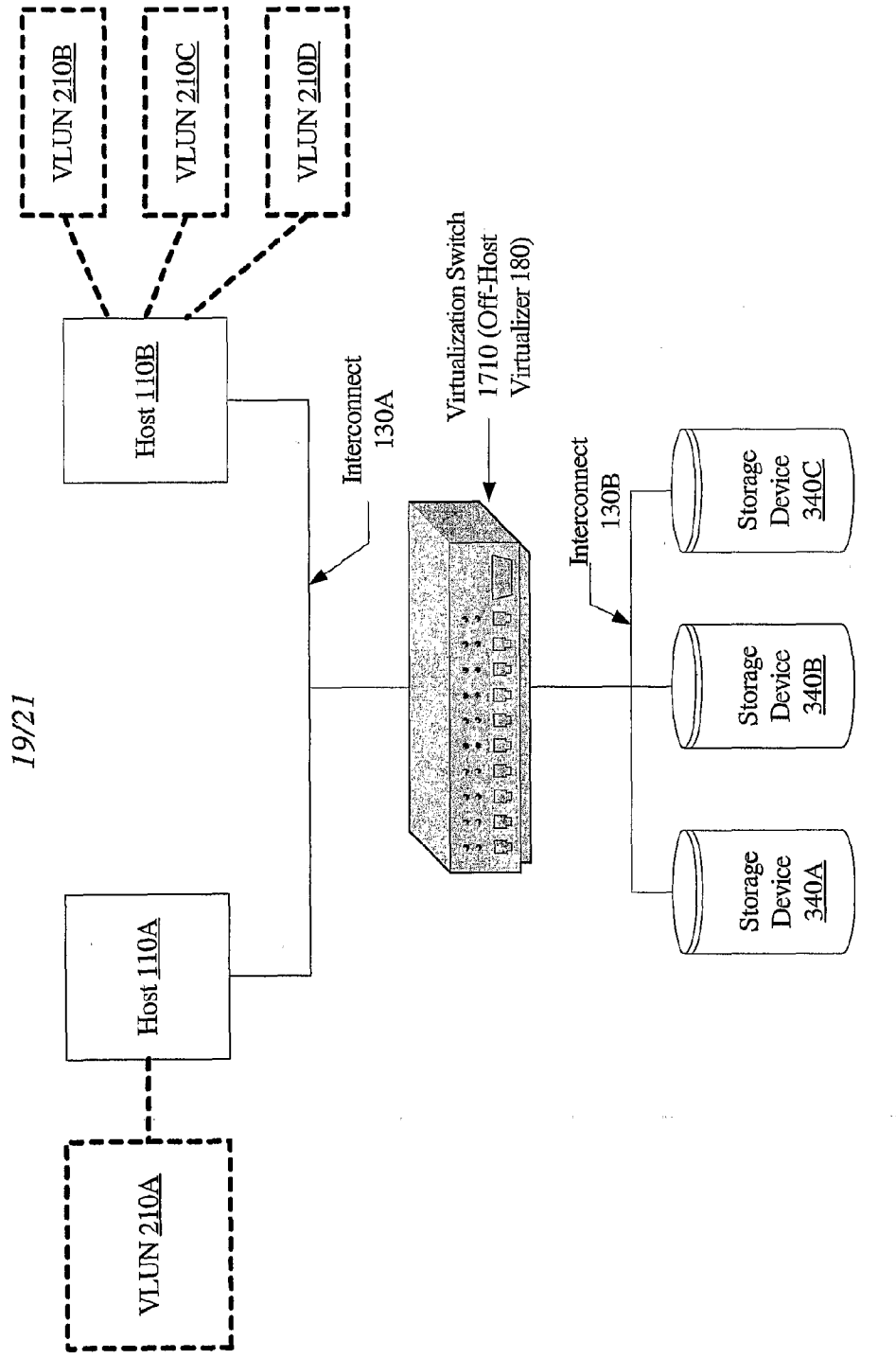


FIG. 17

SAMSUNG EX1002, Page 321 of 1031

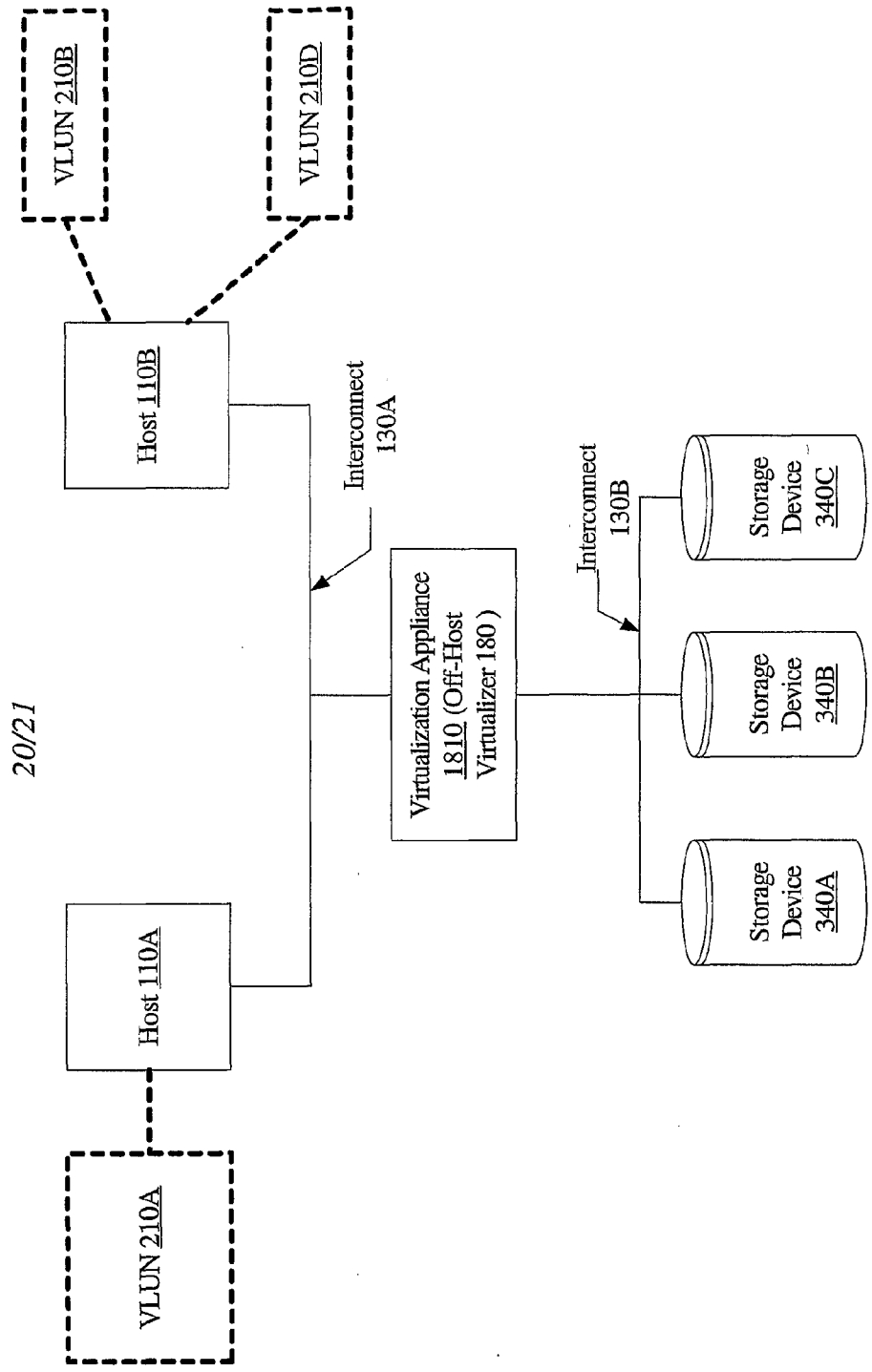


FIG. 18

SAMSUNG EX1002, Page 322 of 1031

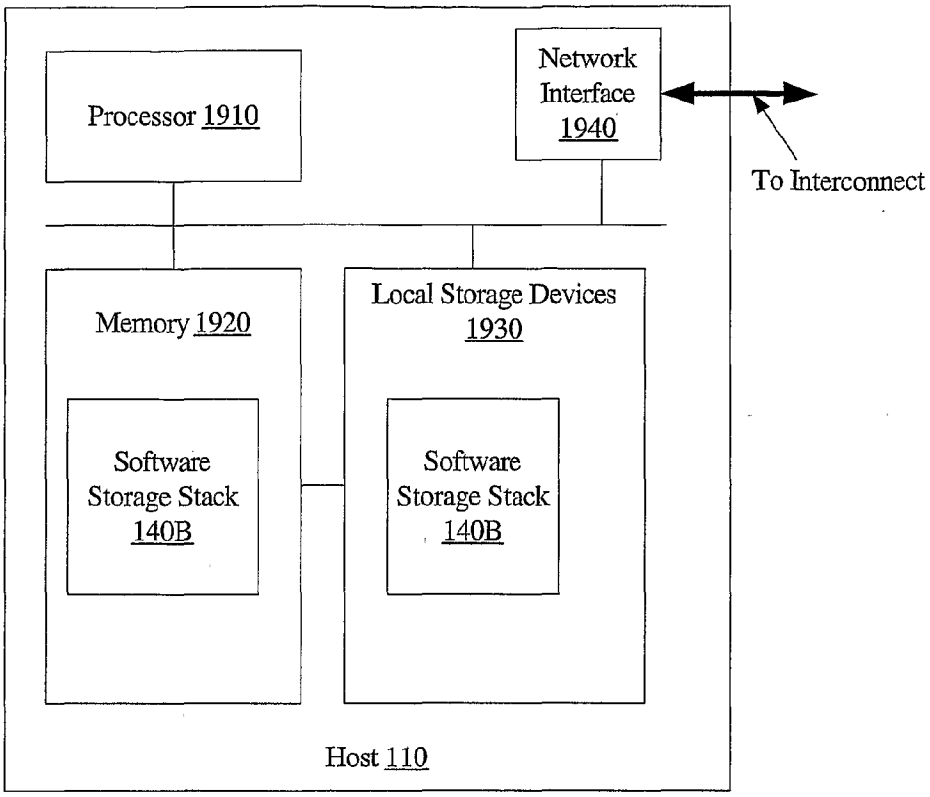


FIG. 19

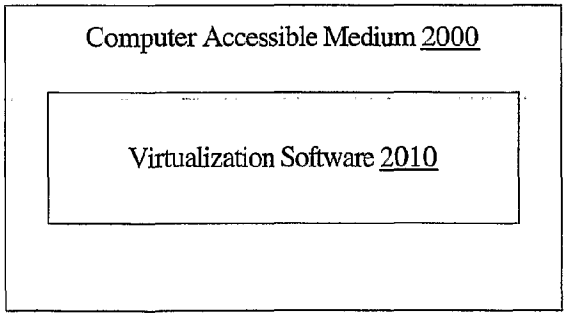


FIG. 20

SAMSUNG EX1002, Page 323 of 1031

INTERNATIONAL SEARCH REPORT

International Application No
PCT/US2004/039306

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 G06F3/06

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)
IPC 7 G06F

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

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

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No
X	ANONYMOUS: "IBM Storage Tank TM. A Distributed Storage System" IBM, 24 January 2002 (2002-01-24), XP002270407 page 2 - page 3 page 5 - page 7	1-8, 20-22, 32,33
Y	page 2 - page 3	9-19, 23-31, 34-36

-/--

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents

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

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

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
10 May 2005	20/05/2005

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx 31 651 epo nl, Fax (+31-70) 340-3016	Dewyn, T

Form PCT/ISA/210 (second sheet) (January 2004)

SAMSUNG EX1002, Page 324 of 1031

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2002/156984 A1 (PADOVANO MICHAEL) 24 October 2002 (2002-10-24) paragraph '0043! - paragraph '0044! paragraph '0070! - paragraph '0077! paragraph '0089! paragraph '0108! paragraph '0123! - paragraph '0125! paragraph '0148!	9-19, 23-31, 34-36
A	US 2003/004972 A1 (WINOKUR ALEXANDER) 2 January 2003 (2003-01-02) the whole document	1-36
A	WO 02/065249 A (CONFLUENCE NETWORKS, INC; GAJJAR, KUMAR; COLLINS, JIM; PATEL, DIPAM; M) 22 August 2002 (2002-08-22) the whole document	1-36
A	US 2002/103889 A1 (MARKSON THOMAS ET AL) 1 August 2002 (2002-08-01) the whole document	1-36
A	US 6 493 811 B1 (BLADES JERRY A ET AL) 10 December 2002 (2002-12-10) the whole document	1-36
A	US 2002/087544 A1 (SELKIRK STEPHEN S ET AL) 4 July 2002 (2002-07-04) the whole document	1-36
A	US 5 829 053 A (SMITH ET AL) 27 October 1998 (1998-10-27) the whole document	1-36

Form PCT/ISA/210 (continuation of second sheet) (January 2004)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No
PCT/US2004/039306

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2002156984 A1	24-10-2002	EP 1382176 A2	21-01-2004
		WO 02067529 A2	29-08-2002
US 2003004972 A1	02-01-2003	WO 03005204 A1	16-01-2003
WO 02065249 A	22-08-2002	EP 1370947 A1	17-12-2003
		EP 1370945 A1	17-12-2003
		EP 1370950 A1	17-12-2003
		EP 1368742 A2	10-12-2003
		JP 2004532442 T	21-10-2004
		JP 2004523831 T	05-08-2004
		WO 02065249 A2	22-08-2002
		WO 02065309 A1	22-08-2002
		WO 02065290 A1	22-08-2002
		WO 02065298 A1	22-08-2002
		US 2002156987 A1	24-10-2002
		US 2002188711 A1	12-12-2002
		US 2002174306 A1	21-11-2002
		US 2003037127 A1	20-02-2003
		US 6880062 B1	12-04-2005
		US 2005027754 A1	03-02-2005
US 2002103889 A1	01-08-2002	US 6779016 B1	17-08-2004
		WO 0198906 A2	27-12-2001
		AT 265707 T	15-05-2004
		AU 769928 B2	12-02-2004
		AU 6918200 A	19-03-2001
		CA 2376333 A1	01-03-2001
		CN 1373871 A	09-10-2002
		DE 60010277 D1	03-06-2004
		DE 60010277 T2	05-01-2005
		EP 1206738 A2	22-05-2002
		JP 2003507817 T	25-02-2003
		TW 548554 B	21-08-2003
		WO 0114987 A2	01-03-2001
		US 6714980 B1	30-03-2004
		US 2003126265 A1	03-07-2003
		US 6597956 B1	22-07-2003
		US 2003154279 A1	14-08-2003
		US 2002052941 A1	02-05-2002
		AU 7361701 A	02-01-2002
US 6493811 B1	10-12-2002	US 2003033497 A1	13-02-2003
US 2002087544 A1	04-07-2002	WO 02054290 A2	11-07-2002
US 5829053 A	27-10-1998	US 5940867 A	17-08-1999

INFORMATION DISCLOSURE STATEMENT

First Inventor: Kuzmin, Andrey V.	**Conf. No.:** 3894
App. No.: 17/931,506	**Filed:** September 12, 2022
Examiner: Bataille, Pierre Miche	**Art Unit:** 2136
Title: STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS	

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

References and Listing

Pursuant to 37 CFR 1.56, an Information Disclosure Statement listing references is provided herewith. Copies of any listed foreign and non-patent literature references are being submitted. Any foreign references may also include English abstract(s) and/or machine translation(s), but no representation is made as to their accuracy.

Pursuant to 37 CFR 1.97(g) and (h), Applicant makes no representation that the information is considered to be material to patentability. Additionally, inclusion on this list is not an admission that any of the cited documents are prior art in this application. Further, Applicant makes no representation regarding the completeness of this list, or that better art does not exist.

If the Examiner would like additional information regarding these references or if anything is unclear, the Examiner is invited to contact the undersigned for assistance.

No Disclaimers

To the extent that anything in the Information Disclosure Statement or the listed references could be construed as a disclaimer of any subject matter supported by the present application, Applicant hereby rescinds and retracts such disclaimer.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /P.B/

Other Applications of Assignee

Applicant wishes to draw the Examiner's attention to, and encourages the Examiner to review, the following co-owned patents and/or applications and their existing and ongoing prosecution history:

Application No.	Filing Date	Patent No.
13/767,723	2/14/2013	9,652,376
14/047,193	10/7/2013	9,229,854
14/951,708	11/25/2015	9,519,578
15/346,641	11/8/2016	9,710,377
15/621,888	6/13/2017	10,445,229
16/570,922	9/13/2019	11,118,457
16/751,925	1/24/2020	11,314,636
16/783,083	2/5/2020	11,334,479
16/794,805	2/19/2020	
16/833,547	3/28/2020	11,216,365
17/563,481	12/28/2021	11,487,657
17/563,467	12/28/2021	11,487,656
17/571,394	1/7/2022	
17/877,881	7/29/2022	
17/931,516	9/12/2022	
17/952,964	9/23/2022	
17/952,230	9/24/2022	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /P.B/

SAMSUNG EX1002, Page 328 of 1031

Applicant notes that cited references, office actions, responses, and notices of allowance currently exist or will exist with reference to the above-referenced matters. Applicant also understands that the Examiner has access to sophisticated Patent Office computing systems that provide ready access to the full file histories of these matters including, for example, specifications, drawings, pending claims, cited art, office actions, responses, declarations, and notices of allowance. Rather than submit copies of these file histories, Applicant respectfully requests that the Examiner continue to review these file histories online for past, current, and future information about these matters that may be relevant to examination of the present application. With respect to parent applications of the present application, Applicant understands that, in accordance with MPEP 609, the Examiner of the present application will consider information which has been considered by the Patent Office in the parent applications. If the Examiner cannot readily access these file histories and information, Applicant would be pleased to provide any portion of any of the file histories or information at any time upon specific Examiner request.

Timing of Disclosure

This Information Disclosure Statement is being filed after receipt of a First Office Action, but before the mailing date of a Final Action and before the mailing date of a Notice of Allowance.

This Statement is accompanied by the fees set forth in 37 CFR 1.17(p). The Commissioner is hereby authorized to charge any additional fees which may be required or to credit any overpayment to Account No. 11-1410.

Respectfully submitted,

KNOBBE, MARTENS, OLSON & BEAR, LLP

Dated: December 13, 2022

By: /Aaron M. Davis/
Aaron M. Davis
Registration No. 62,027
Registered Practitioner
(858) 707-4000

56774878

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /P.B/

SAMSUNG EX1002, Page 329 of 1031

INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Application No.	17/931,506
	Filing Date	September 12, 2022
	First Named Inventor	Kuzmin, Andrey V.
	Art Unit	2136
(Multiple sheets used when necessary)	Examiner	Bataille, Pierre Miche
SHEET 1 OF 2	Attorney Docket No.	RMMS.003C12

U.S. PATENT DOCUMENTS

Examiner Initials	Cite No.	Document Number *Number - Kind Code (if known)* Example: 1,234,567 B1	Publication Date MM-DD-YYYY	Name	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear
	1	5,963,977	10-05-1999	Gold	
	2	6,892,287	05-10-2005	Millard	
	3	7,631,138	12-08-2009	Gonzalez	
	4	8,341,339	12-25-2012	Boyle	
	5	11,307,995	04-19-2022	Lercari	
	6	11,487,656	11-01-2022	Kuzmin et al.	
	7	11,487,657	11-01-2022	Kuzmin et al.	
	8	2006/0004957	01-05-2006	Hand III	
	9	2011/0041039	02-17-2011	Harari	
	10	2011/0238892	09-29-2011	Tsai	
	11	2011/0238943	09-29-2011	Devendran et al.	
	12	2011/0264843	10-27-2011	Haines	
	13	2012/0155492	06-21-2012	Abel	
	14	2012/0198128	08-02-2012	Van Aken	
	15	2012/0303875	11-29-2012	Benhase	
	16	2013/0097236	04-18-2013	Khorashadi	
	17	2013/0166824	06-27-2013	Shim	

FOREIGN PATENT DOCUMENTS

Examiner Initials	Cite No.	Foreign Patent Document *Country Code-Number-Kind Code* Example: JP 1234567 A1	Publication Date MM-DD-YYYY	Name	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear	T[1]
	18	JP 2001051889	02-23-2001	Sharp Corp.		X
	19	JP 2011123863	06-23-2011	Phison Electronics Corp.		X
	20	JP 2011203916	10-13-2011	Toshiba Corp.		X
	21	KR 20000025612	05-06-2000	LG Electronics Inc.		X
	22	RD 527033	03-10-2008			

Examiner Signature	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 next communication to applicant.

T[1] - Place a check mark in this area when an English language Translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /P.B/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Application No.	17/931,506
	Filing Date	September 12, 2022
	First Named Inventor	Kuzmin, Andrey V.
	Art Unit	2136
(Multiple sheets used when necessary)	Examiner	Bataille, Pierre Miche
SHEET 2 OF 2	Attorney Docket No.	RMMS.003C12

NON PATENT LITERATURE DOCUMENTS			
Examiner Initials	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	T[1]
	23	Y. H. Chang et al. "An efficient FTL design for multi-chipped solid-state drives," 2010 IEEE 16th intl. conference on embedded and real-time computing systems and applications, 2010, pp. 237-246.	
	24	Eric Seppanen, Matthew T. O'Keefe, David J. Lilja, Department of Electrical and Computer Engineering, University of Minnesota, ""High Performance Solid State Storage Under Linux,"" 4-10-2010, MSST '10 Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), Pages 1-12.	

56774681

Examiner Signature	/PIERRE MICHEL BATAILLE/	Date Considered	12/20/2022

*__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 next communication to applicant.

T[1] - Place a check mark in this area when an English language Translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /P.B/

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
15 May 2008 (15.05.2008)

PCT

(10) International Publication Number
WO 2008/057557 A2

(51) **International Patent Classification:** **Not classified**

(21) **International Application Number:**
PCT/US2007/023432

(22) **International Filing Date:**
6 November 2007 (06.11.2007)

(25) **Filing Language:** English

(26) **Publication Language:** English

(30) **Priority Data:**
60/857,408 6 November 2006 (06.11.2006) US

(71) **Applicant** *(for all designated States except US):* **RAMBUS INC.** [US/US]; 4440 El Camino Real, Los Altos, CA 94022 (US).

(72) **Inventors; and**

(75) **Inventors/Applicants** *(for US only):* **WARE, Frederick, A.** [US/US]; 13961 Fremont Pines Lane, Los Altos Hills, CA 94022 (US). **TSERN, Ely, K.** [US/US]; 784 Dixon Way, Los Altos, CA 94022 (US).

(74) **Agent: BEHIEL, Arthur, J.;** Silicon Edge Law Group LLP, 6601 Koll Center Parkway, Suite 245, Pleasanton, CA 94566 (US).

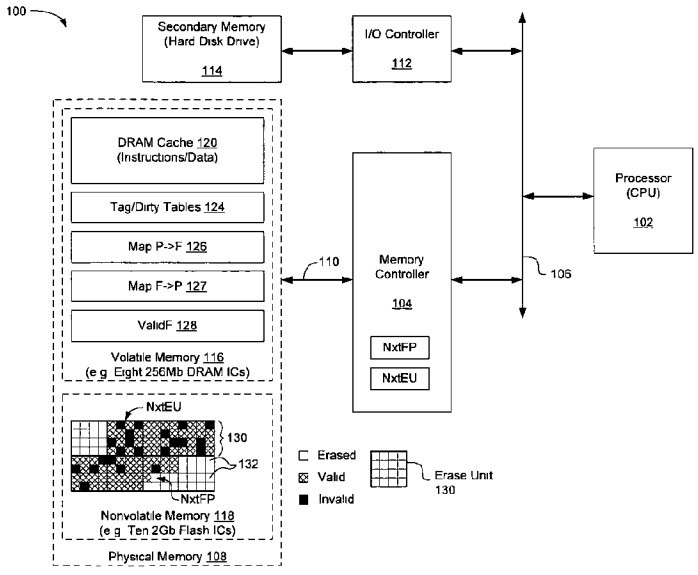
(81) **Designated States** *(unless otherwise indicated, for every kind of national protection available):* AE, AG, AL, AM, **AT, AU, AZ, BA, BB,** BG, **BH, BR,** BW, **BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC,** EE, **EG,** ES, **FI, GB, GD,** GE, GH, GM, GT, **HN, HR, HU, ID,** IL,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

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available):* ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, **DK,** EE, ES, FI, FR, GB, GR, HU, IE, **IS, IT,** LT, LU, LV, MC, MT, NL, PL, PT, **RO,** SE, **SI,** SK, TR), OAPI (BF, **BJ,** CF, CG, CI, CM, GA, GN, GQ, GW, ML, **MR,** NE, SN, TD, TG).

Declaration under Rule 4.17:
— *of inventorship (Rule 4.17(iv))*

Published:
— *without international search report and to be republished upon receipt of that report*

(54) **Title:** MEMORY SYSTEM SUPPORTING NONVOLATILE PHYSICAL MEMORY



(57) **Abstract:** A memory system includes nonvolatile physical memory, such as flash memory, that exhibits a wear mechanism asymmetrically associated with write operations. A relatively small cache of volatile memory reduces the number of writes, and wear-leveling memory access methods distribute writes evenly over the nonvolatile memory.

WO 2008/057557 A2

MEMORY SYSTEM SUPPORTING NONVOLATILE PHYSICAL MEMORY

FIELD

[0001] The disclosed embodiments relate generally to memory systems and methods, and in particular to non-homogeneous memory systems and methods.

BACKGROUND

[0002] Computer program code and data needed for execution of a process on a computer system typically reside in the computer system's physical memory. Sometimes referred to as "primary storage," "internal memory," or "main memory," physical memory is accessible to the computer's central processing unit (CPU) without the use of the computer's input/output (I/O) channels. Physical memory typically affords very fast access, and is therefore frequently used to store information that is or is likely to be in active use by the CPU.

[0003] Because physical memory is typically volatile and relatively expensive, many computer systems utilize secondary storage. This secondary storage is usually accessible to the CPU via the computer's I/O channels and is normally used for persistent storage of relatively large amounts of data. Since the physical memory in many computer systems is more limited than what would be required to store all the program code and data that a user may want to access, many computer systems have what is called "virtual memory."

[0004] In a computer system with virtual memory, some of the program code and/or data in the physical memory may be removed from physical memory and stored in the secondary storage, such as the computer's hard disk, until space becomes available on the physical memory or until such code or data is needed by the CPU. Because access to the code and

12/21/2022 19:01:46 Page 2 of 31
SAMSUNG EX1002, Page 333 of 1031

data in secondary storage is typically slower access than from physical memory, it is desirable to swap code or data in the physical memory that is unlikely to be used soon by the CPU with code or data in the secondary storage.

[0005] Many computer systems employing virtual memory have automatic storage allocation schemes to manage the transfer of code and storage between physical memory and secondary storage. Pages of data and program code are transferred from secondary storage to physical memory as the data and program code is needed by an executing process, and pages of data and program code are evicted from physical memory and written to secondary storage when room is needed in physical memory to store other pages of data and program code. The process of moving pages of data and program code back and forth between physical memory and secondary storage is called by a variety of names, including swapping, paging, and virtual memory management.

[0006] The location in the physical memory where a particular piece of data or code is stored is referred to as the physical address. A logical address or virtual address is a memory location accessed by an application program in a system with virtual memory such that intervening computer hardware and/or software maps the virtual address to physical memory. During the course of execution of the application, the same virtual address may be mapped to many different physical addresses as data and programs are paged out and paged in to other locations. The virtual address (which may also be referred to as the logical address) typically includes a virtual (or logical) page number plus the location within that page, and is typically interpreted or mapped onto a physical memory address by the operating system using an address translation function. If the page is present in physical memory, the address translation function substitutes the physical page frame number for the logical number. If the address translation function detects that the page requested is not present in physical memory,

12/21/2022 19:01:46 Page 3 of 31
SAMSUNG EX1002, Page 334 of 1031

a fault occurs and the page is read into a physical memory page frame from secondary storage. This address translation function can be accomplished using a directly indexed table, commonly referred to as a "page table," which identifies the location of the program's pages in physical memory. If the page table indicates that a page is not resident in physical memory, the address translation function issues a page fault to the operating system. This causes execution of the program which required the page to be suspended until the desired page can be read from secondary storage and placed in physical memory.

[0007] Portable computing devices typically use a single type of memory device at each level in their memory hierarchy. Higher levels in the memory hierarchy typically operate at higher speeds and lower latency than lower levels. For example, portable computers (e.g., notebook computers) typically have three or more hierarchical levels of memory, including cache memory, secondary storage, and physical memory. Cache memory is typically implemented using Static Random Access Memory (SRAM). Often there are two or more levels of cache memory. Secondary storage is typically implemented with magnetic disk storage (often called hard disk storage). Physical memory is typically implemented with Dynamic Random Access Memory (DRAM). In some portable computers, such as personal digital assistants (PDAs), the secondary storage is implemented using flash memory instead of magnetic disk storage.

[0008] DRAM has a near-optimal combination of operational attributes for implementing physical memory. These attributes include, without limitation, low cost (only magnetic disk storage has a lower per-bit cost), low read time (the read access time is within an order of magnitude of that of the highest speed SRAM), low write time that is the same or similar to the read access time, and essentially unlimited endurance (i.e., the storage cell can be rewritten an unlimited number of times). Unfortunately, DRAM is volatile, which is to say

12/21/2022 19:01:46 Page 4 of 31
SAMSUNG EX1002, Page 335 of 1031

that DRAM cannot maintain stored data in the absence of applied power. This means that computers relying upon DRAM for physical memory must maintain applied power even when not in use—a problem of particular significance for battery-powered devices—or require slow boot processes during which information is moved from secondary memory into physical memory. There is therefore a need for memory systems that include nonvolatile physical memory.

BRJEF DESCRIPTION OF THE DRAWINGS

[0009] The subject matter disclosed is illustrated by way of example, and not by way of limitation, in the figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

[0010] Figure 1 depicts a portion of a computer system 100 that employs nonvolatile physical memory in accordance with one embodiment.

[0011] Figure 2 is a flowchart 200 depicting the operation of computer system 100 of Figure 1 in accordance with one embodiment.

[0012] Figure 3 depicts a memory system 300 in accordance with another embodiment, including a memory controller 305, a DRAM cache 310 for instructions and data, DRAM tag and map tables 3 15, and flash memory 320.

[0013] Figure 4 depicts an addressing scheme that may be used in an embodiment of memory system 300 of Figure 3.

[0014] Figure 5 is a flowchart 500 depicting the operation of the embodiment of memory system 300 of Figure 3 in terms of the addressing scheme of Figure 4.

[0015] Figure **6 is** a flowchart 600 illustrating a garbage-collection process implemented by controller 305 of Figure **3** in one embodiment.

12/21/2022 19:01:46 Page 5 of 31
SAMSUNG EX1002, Page 336 of 1031

DETAILED DESCRIPTION

[0016] Figure 1 depicts a portion of a computer system 100 that employs nonvolatile physical memory in accordance with one embodiment. Computer system 100 includes a processor or central processing unit (CPU) 102 connected to a memory controller 104 via a system bus 106. Memory controller 104 acts as a bridge between CPU 102 and some physical memory 108. Other terms are often used in the computer industry to describe this type of bridge device, including "north bridge" or "memory controller hub." Memory controllers are often sold as part of a set of devices, commonly referred to as the system "chip set," though memory control functions can also be supported with other functions on a common integrated circuit (IC) device, including the CPU itself, and/or can be distributed among a number of devices.

[0017] Memory controller 104 is connected to a physical memory 108 via a communication channel 110 and to an IO controller 112 via system bus 106, though the memory and IO controllers may connect to different buses in other embodiments. IO controller 112 interfaces to peripheral devices, including e.g. a hard disk drive 114. Disk drive 114 serves as secondary storage in the present example. System bus 106 and channel 110 may include one or more signal lines, such as, for example signal lines to carry address, data, control, strobe, or clock signals. These details are readily understood by those of skill in the art, and are omitted here for brevity.

[0018] Memory controller 104 may translate physical addresses P from CPU 102 to an addressing scheme suitable for physical memory 108. Such translation allows memory controller 104 to support different physical-memory configurations. In the instant example, however, physical addresses P from CPU 102 are assumed to be suitable for use with

12/21/2022 19:01:46 Page 6 of 31
SAMSUNG EX1002, Page 337 of 1031

physical memory 108 without translation. Memory controller 104 additionally includes a pair of registers NxtFP and NxtEU, the purposes of which are detailed below.

[0019] Physical memory 108 includes volatile and nonvolatile portions 116 and 118, DRAM and flash memory in this embodiment. Volatile and nonvolatile portions 116 and 118 each store data in terms of "access units," which may be defined as the smallest memory areas handled as a single chunk of data. In the instant example, volatile memory 116 supports volatile access units that store two-kilobyte pages, and nonvolatile memory 118 supports nonvolatile access units that store pages of the same size. The volatile and nonvolatile access units may be sized differently in other embodiments.

[0020] Reads and writes to nonvolatile memory 118 may be performed one access unit at a time, in a random-access fashion, but erasures are carried out on collections of access units. Flash memory 118 is therefore divided into erase units 130, sometimes called "erase blocks," each of which is divided into nonvolatile access units 132. Each nonvolatile access unit 132 within an erased erase unit 130 can be written to or read from. Once written to, however, a nonvolatile access unit 132 cannot be written to again until the entire erase unit 130 is erased.

[0021] DRAM 116, at the direction of memory controller 104, provides a cache 120 for instructions and data and tag and dirty tables 124. CPU 102 employs tables 124 to keep track of which virtual pages reside in physical memory 108 and whether those pages have been written to without the changes having yet been saved to a lower level in the memory hierarchy (i.e., are dirty). Virtual-to-physical and physical-to-virtual address translation tables (not shown) may be held in secondary memory, and may be moved to physical memory by paging software (also not shown) executing on CPU 102. These and other details that relate to the use of virtual memory are well understood by those of skill in the art, and are therefore omitted for brevity.

12/21/2022 19:01:46 Page 7 of 31
SAMSUNG EX1002, Page 338 of 1031

[0022] Controller 104 tracks dirty pages in DRAM 116, which are those pages that include changes not reflected in corresponding memory locations within flash memory 118. In accordance with the depicted embodiment, controller 104 also uses DRAM 116 to store maps 126 and 127 of physical-to-flash (P->F) and flash-to-physical (F->P) address translations identifying where data and instructions in cache 120 have corresponding pages in flash memory 118 and *vice versa,* and to store a valid table 128 identifying which pages in flash memory 118 include valid data. Flash memory 118 will typically include considerably more address locations than will DRAM 116. For example, volatile portion 116 may include eight 256Mb DRAM ICs and nonvolatile portion 118 may include ten 2Gb flash memory ICs in one embodiment.

[0023] In the example of Figure 1, CPU 102 accesses (reads or writes) cache lines of e.g. sixty-four bytes from DRAM 116 via controller 104. Memory controller 104 fills volatile memory 116 in volatile access units (e.g. DRAM pages) and fills nonvolatile memory 118 in nonvolatile access units (e.g. Flash pages) of the same size as the volatile access units. With reference to the key in the lower center of Figure 1, access units 132 can be erased or can contain information that is either valid or invalid. These distinctions are described below.

[0024] Figure 2 is a flowchart 200 depicting the operation of portions of computer system 100 of Figure 1 in accordance with one embodiment. The process begins when CPU 102 presents a physical cache line address to memory controller 104 as part of the process of requesting access to a page (step 205). Memory controller 104 determines whether the requested page is in DRAM cache 120 (decision 215). If so, and if the access request is to read data (decision 220), then memory controller 104 reads the requested line from cache 120 and presents the resulting information to CPU 102 via system bus 106 (step 225). If the information is in cache 120 and the access request is to write data, then memory controller

12/21/2022 19:01:46 Page 8 of 31
SAMSUNG EX1002, Page 339 of 1031

104 writes data from CPU 102 into the addressed line of cache 120 (step 230) and sets the dirty bit associated with the written page address in cache 120 (step 235). The dirty bit marks the fact that the page in cache 120 has been changed, and so cannot be flushed from cache 120 or overwritten unless the modified page is copied to a lower level in the memory hierarchy. Decision 215 relies upon entries in tables 124 that indicate which pages, by physical page address, reside in cache 120.

[0025] Returning to decision 215, if the requested page is not in cache 120, then memory controller 104 copies the requested page from flash memory 118 into cache 120 in preparation for the requested memory access. To do this, memory controller 104, using the contents of map 126, translates the requested physical address P into the corresponding flash address F (step 240). Before copying the selected flash page into the DRAM cache, memory controller 104 determines whether the page to be overwritten in cache 120 (the target page) represents a "dirty" page by referencing tables 124 in DRAM 116 that memory controller 104 maintains for this purpose.

[0026] If that target page is not dirty, per decision 245, then the contents of the requested page is loaded from flash memory 118 into the clean page in cache 120 (step 247). If the target page is dirty, however, before loading the requested page from flash memory 118 to cache 120, memory controller 104 copies the target page to flash memory 118 to retain whatever changes were made to the target page since the target page was loaded from flash memory 118. Memory controller 104 (1) loads the dirty target page from cache 120 into an address in flash memory 118 identified as the next flash page by register NxtFP (step 250), (2) marks the prior flash address associated with the dirty page as invalid in table 128 (step 255), and (3) changes the physical-to-flash mapping associated with the dirty page so that subsequent requests for the newly saved page will access the updated flash page (step 260).

12/21/2022 19:01:46 Page 9 of 31
SAMSUNG EX1002, Page 340 of 1031

With the contents of the target page in cache 120 thus preserved, the contents of the requested page is loaded from flash memory 118 into the formerly dirty page in cache 120 and the corresponding dirty bit in table 124 is reset (step 265). The next-flash page address NxtFP is then changed to point to an erased page 132 in flash memory 118 in preparation for the next write to flash memory (step 270). The process then returns to decision 220 for completion of the memory access.

[0027] Some embodiments favor overwrites to clean pages over dirty ones to reduce the overhead and lifespan reduction associated with a write. Using a multi-set DRAM cache, for example, the selection of a page to overwrite could factor in the clean or dirty state of the target page. The process might also take other factors into account when selecting a target page. Some embodiments, for example, favor writes to the least-recently used clean location.

[0028] The number of flash pages (e.g., access units 132 of Figure 1) marked as invalid will increase over time. A garbage collection process may therefore be performed from time to time to recover invalid pages for subsequent use. In one embodiment, register NxtEU may be implemented as a simple counter that identifies the address of the next one of erase blocks 130. When the number of erased memory pages drops below a threshold, for example, memory controller 104 may copy each of the valid pages in the next erase block into available erased flash pages before erasing the erase block and changing the contents of register NxtEB to point to the next candidate for an erase operation. In some embodiments memory controller 104 maintains a table in physical memory 108 that keeps track of the number of invalid pages in each erase block 130. When the number of erased pages falls below some threshold, an erase block with many or the most invalid pages may be erased.

[0029] Figure 3 depicts a memory system 300 in accordance with another embodiment, including a memory controller 305, volatile DRAM memory 307, and nonvolatile flash

12/21/2022 19:01:46 Page 10 of 31
SAMSUNG EX1002, Page 341 of 1031

memory 320. Flash memory 320 is divided up into groups of nonvolatile access units (pages) 340 and erase units 345. Volatile memory 307 includes a DRAM cache 310 and DRAM tag and map tables 315. Cache 310 stored instructions and data for the CPU (not shown), and may be similar to cache 120 of Figure 1. Tag and map tables 315, in this embodiment, includes:

1. a tag/dirty/valid table 325 that identifies which physical address locations in cache 310 include pages of valid information, and which of those represents dirty pages;
2. a mapPF table 330 that memory controller 305 may use to maintain a mapping between physical addresses and flash addresses for pages 340; and
3. a MapFP/ValidF table 335 that memory controller 305 may use to maintain a mapping between flash addresses and physical addresses and to identify valid and invalid page entries in flash memory 320.

[0030] Memory controller 305 includes two registers that keep track of the amount of available erased pages in flash memory 320: a head register HeadF contains the address of the next empty one of flash pages 340 to receive data writes, and a tail register TailF contains the address of the one of erase units 345 storing the eldest data. The erase unit with the eldest data is likely to be among the erase units with the highest number of invalid page entries. Erasing the erase unit with the eldest data is therefore likely to free up a relatively large number of nonvolatile access units for future writes. Memory controller 305 communicates with the other components of memory system 300 over a number of ports, descriptions of some of those ports are provided below in connection with Figures 4-6.

[0031] Figure 4 depicts an addressing scheme that may be used in an embodiment of memory system 300 of Figure 3. Figure 5 is a flowchart 500 depicting the operation of that

SAMSUNG EX1002, Page 342 of 1031

embodiment in terms of the addressing scheme of Figure 4. On the software side of Figure 4, the CPU expresses 36b (thirty-six bit) virtual byte addresses 405, each including a 25b page address and an 11b page offset. The CPU converts virtual byte addresses 405 to 31b physical byte addresses 410 using address translation tables maintained in a translation look-aside buffer (TLB) translation table (not shown) within the CPU. The TLB is a type of cache that holds copies of the virtual-to-physical translations for the most recently used pages. The specified numbers of bits for the various fields are exemplary and not limiting, as will be readily understood by those of skill in the art.

[0032] In the example of Figure 4, each physical byte address P 410 includes a 20b physical page address and an 11b page offset. The page address is divided into a 3b field PH and a 17b field PM. The page offset contains a 5b field PL for specifying a 64 byte line in the page. In this example, it is assumed that a read or write memory access transfers a 64 byte line. The CPU and translation table are omitted from Figure 3, but examples are discussed above in connection with Figure 1.

[0033] In one embodiment, the twenty-five most significant bits of physical byte address P 410 represent a physical line address (PH:PM:PL), which may be logically divided into the three most-significant bits PH, the middle seventeen bits PM, and the least-significant five bits PL. The tag portion of tables 325 includes columns of three-bit tags (aligned to the PH field) each associated with a 17-bit physical tag address (aligned to the PM field). Flash byte addresses 425 are expressed as a 21b flash page address and an 11b page offset. The flash page address is divided into a 15b field FH and a 5b field FM.

[0034] Flowchart 500 of Figure 5 outlines how memory controller 305 of Figure 3 may perform read and write accesses for pages located in physical memory. As in the prior example of Figure 1, the physical memory in system 300 of Figure 3 includes DRAM and

12/21/2022 19:01:46 Page 12 of 31
SAMSUNG EX1002, Page 343 of 1031

flash memory. The process of Figure 5 commences when memory controller 305 attempts to access a physical memory location specified by a physical byte address P 410 (step 505). At decision 510, memory controller 305 examines the contents of tables 325 to determine whether the requested address is in DRAM cache 310. To do this, memory controller 305 accesses the tag memory at the address location identified by the middle bits PM of physical byte address 410. A match between the three most significant bits PH of physical byte address 410 and the three-bit tag of the selected tag address in tag memory indicates a cache hit.

[0035] In the case of a cache hit, memory controller 305 either reads from or writes to the location in DRAM cache 310 identified by the mid- and low-order bit fields PM:PL of physical byte address 410 (decision 515). For a read, memory controller 305 causes DRAM cache 310 to present the contents of the addressed cache line on data signals DQ (step 520) and sends the read data to the CPU. For a write, memory controller 305 causes DRAM cache 310 to accept data provided from the CPU onto data signals DQ into the addressed cache line (step 525). Memory controller 305 then alters the contents of tables 325 to mark the altered page in cache 310 as dirty (step 530).

[0036] In the case of a cache miss, memory controller 305 will move the requested page from flash memory 320 into an address location in DRAM cache 310. This "target" address location in cache 310 will typically be the least-recently used (accessed) page. Memory controller 305 considers tables 325 to determine whether the target page is dirty (decision 535). If the target page is clean, an identical copy of the target page resides in flash memory 345; if dirty, then the corresponding copy in flash memory 345 is stale, so the dirty page is written to flash memory 345 to preserve the information in the target page.

12/21/2022 19:01:46 Page 13 of 31
SAMSUNG EX1002, Page 344 of 1031

[0037] If controller 305 determines that the target page is dirty, then the target page at DRAM cache address PM is loaded into flash memory 345 at the address of the erased page specified by register HeadF (step 555). Table 330 is then updated to reflect the new physical-to-flash mapping between the target page in cache 310 and the flash-head address HeadF (step 560) using the PH:PM fields of the physical address. The flash page address formerly associated with the target page in cache 310 is marked invalid in tables 335 (step 565) using the PH:PM fields of the physical address. The flash head page address HeadF is loaded into table 330 to update the flash-to-physical mapping for the target page (step 570) using the PH//PM fields of the physical address, and address HeadF is marked as valid (step 571). Controller 305 then increments head address HeadF (step 575). The process then moves to step 540.

[0038] For clean target pages, memory controller 305, using physical-to-flash mapping tables 330, converts the requested physical page address PH:PM into a flash page address FH:FM and loads the contents of that flash page into the target page location PM in cache 310 (step 540). The tag location at PM is loaded with the upper physical address field PH (step 545), and the dirty location at PM is cleared to zero (step 580). The process then moves to decision 515 and the requested memory access, either a read or a write, is completed.

[0039] The process of flowchart 500 is repeated for each access to physical memory. Each cache miss to a dirty page produces an invalid page 340 in a page in flash memory 345. If allowed to accumulate, the invalid page entries would prevent memory controller 305 from writing new pages into flash memory 320. Memory controller 305 therefore supports a garbage-collection process that recycles invalid pages.

[0040] Figure 6 is a flowchart 600 illustrating a garbage-collection process which may be implemented by controller 305 of Figure 3 in one embodiment. If the difference between the

12/21/2022 19:01:46 Page 14 of 31
SAMSUNG EX1002, Page 345 of 1031

head address HeadF and the tail address TailF exceeds some threshold, indicating that flash memory 345 is running low on erased pages, controller 305 begins a garbage collection process (decision 605). Controller 305 then checks tables 335 to determine whether the current tail page contains valid data (decision 610).

[0041] If the tail page contains valid information, the contents of flash memory 345 at address TailF is loaded into address HeadF (step 615) and the flash-to-physical and physical-to-flash mappings in tables 335 and 330 are updated accordingly (steps 620 and 625). Controller 305 then accesses tables 335 to set the valid bit associated with the head address (step 630) and to reset the valid bit associated with the tail address (step 635). The head pointer is then updated by incrementing register HeadF (step 640). Per decision 650, if bits of field FM of TailF are all logic ones, then the tail address corresponds to the last page entry in the selected erase unit. If so, then the erase unit specified by address TaUF[FH] is erased (step 660), the tail pointer is incremented to the first page address in the next erase block (step 665), and the process returns to decision 605. If bits of field FM are not all logic ones, then the tail pointer is incremented (step 655) and the process returns to decision 610. There is no reason to save the contents of invalid flash pages, so memory controller skips from decision 610 to decision 650 when it encounters an invalid page.

[0042] The embodiment of Figure 3 uses a single data structure comprised of a contiguous group of valid and invalid page entries snaking through flash memory 345 to distribute write operations relatively evenly across the flash memory, and to consequently prolong the service life of the flash memory 320. The concept of distributing writes across nonvolatile media is known as "wear leveling," and has been applied to hard drives and solid-state disks that employ flash memory. The system of Figure 3 differs from previous applications in that the remapping of physical to flash addresses is being performed across

12/21/2022 19:01:46 Page 15 of 31
SAMSUNG EX1002, Page 346 of 1031

two of more flash devices by an external controller, and the mapping is a completely arbitrary one (any physical address may be mapped to any flash address without limit). This permits every flash address to be written and erased an equal number of times, maximizing the endurance of the whole flash system.

[0043] Some embodiments can use more than one data structure and/or different kinds of structures. For example, two data structures can work their way through flash memory simultaneously, one for read-only or read-mostly pages and the other for write-often pages. The write-often data structure would be smaller and faster moving, and could jump through and around the other data structure.

[0044] In the above-described examples the access units in physical memory are of the same size as the pages addressed by the CPUs. In other embodiments the life of the nonvolatile memory can be extended by reducing the size of the access units relative to the pages addressed by the CPU. The use of relatively smaller nonvolatile access units reduces the speed at which erase units are filled, and consequently reduces the frequency of erase operations. The tables employed to map requested physical addresses to the relatively small nonvolatile access units would be more complex than in the foregoing embodiments, but the operation of such a memory system would be conceptually similar. The resulting larger tables could be stored in flash memory and cached in DRAM. Increasing the amount of flash memory for implementing a given quantity of physical memory similarly extends the life expectancy of the flash memory.

[0045] In some embodiments the memory controller can support error detection and correction for memory reads. In detecting errors, such embodiments can identify defective flash pages and substitute defective flash pages with spare pages reserved for this purpose. A

12/21/2022 19:01:46 Page 16 of 31
SAMSUNG EX1002, Page 347 of 1031

table distinguishing good and bad flash pages can be maintained in e.g. flash memory for this purpose.

[0046] Flash memory devices are rated based upon a number of performance criteria, including read, write, and erase times, a maximum number of write cycles (endurance count), and the minimum time the memory can be expected to retain stored data (retention time). Flash memory used in the foregoing embodiments might be adapted to favor access times and/or write-cycle endurance over data retention. Depending upon the optimization, the flash memory might require periodic refreshing (e.g., each flash access unit is written to at least once a month), a process that could be supported by the flash device or in an associated memory controller.

[0047] An output of a process for designing an integrated circuit, or a portion of an integrated circuit, comprising one or more of the circuits described herein may be a computer-readable medium such as, for example, a magnetic tape or an optical or magnetic disk. The computer-readable medium may be encoded with data structures or other information describing circuitry that may be physically instantiated as an integrated circuit or portion of an integrated circuit. Although various formats may be used for such encoding, these data structures are commonly written in Caltech Intermediate Format (CIF), Calma GDS II Stream Format (GDSII), or Electronic Design Interchange Format (EDIF). Those of skill in the art of integrated circuit design can develop such data structures from schematic diagrams of the type detailed above and the corresponding descriptions and encode the data structures on computer readable medium. Those of skill in the art of integrated circuit fabrication can use such encoded data to fabricate integrated circuits comprising one or more of the circuits described herein.

12/21/2022 19:01:46 Page 17 of 31
SAMSUNG EX1002, Page 348 of 1031

[0048] While the present invention has been described in connection with specific embodiments, variations of these embodiments will be obvious to those of ordinary skill in the art. For example:

1. The examples provided above employ a direct-map DRAM cache, but other embodiments can employ different cache management schemes (e.g., an N-way associative cache management scheme) for improved performance.
2. The volatile memory can be implemented using technologies other than DRAM.
3. The nonvolatile memory can be implemented using technologies other than flash memory. In general, the memory access methods and circuits are particularly important for memory systems with nonvolatile physical memory that suffers a wear mechanism asymmetrically associated with write operations, but may be employed in other systems as well.
4. The memory controller, two or more of the RAM, the CPU, the memory, and a graphics controller can be instantiated together on the same integrated circuit device or on various combinations of devices.
5. The flowcharts described above are intended to convey an operational sense of some embodiments; in practice, the process flow need not proceed as specified.

Moreover, some components are shown directly connected to one another while others are shown connected via intermediate components. In each instance the method of interconnection, or "coupling," establishes some desired electrical communication between two or more circuit nodes, or terminals. Such coupling may often be accomplished using a number of circuit configurations, as will be understood by those of skill in the art. Therefore,

12/21/2022 19:01:46 Page 18 of 31
SAMSUNG EX1002, Page 349 of 1031

the spirit and scope of the appended claims should not be limited to the foregoing description. Only those claims specifically reciting "means for" or "step for" should be construed in the manner required under the sixth paragraph of 35 U.S.C. §112.

12/21/2022 19:01:46 Page 19 of 31
SAMSUNG EX1002, Page 350 of 1031

CLAIMS

What is claimed is:

1. A memory system:
   a. nonvolatile memory having a plurality of erase units, each erase unit including a plurality of nonvolatile access units, each nonvolatile access unit specified by a respective nonvolatile-access-unit address;
   b. random-access memory (RAM) having a plurality of volatile access units, each volatile access unit specified by a respective volatile-access-unit address, the RAM to store an address translation table to map volatile-access-unit addresses to nonvolatile-access-unit addresses; and
   c. a memory controller having a first data bus coupled to the nonvolatile memory, a second data bus coupled to the RAM, and a next-access-unit register to store the nonvolatile-access-unit address of a next one of the nonvolatile access units to receive write data.

2. The memory system of claim 1, wherein the memory controller includes an erase-unit register to store a block address of a next one of the erase units to erase.

3. The memory system of claim 1, the RAM to further store a second address translation table mapping nonvolatile-access-unit addresses to volatile-access-unit addresses.

4. The memory system of claim 1, further comprising a processor coupled to the memory controller via a system bus.

12/21/2022 19:01:46 Page 20 of 31
SAMSUNG EX1002, Page 351 of 1031

5. The memory system of claim 4, wherein the processor requests instructions and data from the memory controller, and wherein the instructions and data are stored in the RAM and the nonvolatile memory.

6. The memory system of claim 5, wherein the RAM caches the instructions and data stored in the nonvolatile memory.

7. The memory system of claim 5, wherein the processor addresses pages that are sized differently than the nonvolatile access units.

8. The memory system of claim 5, wherein the process addresses pages that are larger than the nonvolatile access units.

9. The memory system of claim 1, wherein at least one of the nonvolatile memory and the RAM further store table entries that identify ones of the nonvolatile access units that include valid data and others of the nonvolatile access units that include invalid data.

10. The memory system of claim 1, wherein the memory controller alters the contents of the next-access unit register after writing data from one of the volatile access units of the RAM to one of the nonvolatile access units in the nonvolatile memory.

11. The memory system of claim 1, wherein the memory controller includes an erase-unit register to store a block address of a next one of the erase units to erase, and wherein the memory controller is alters the contents of the erase-unit register after erasing a specified erase unit.

12/21/2022 19:01:46 Page 21 of 31
SAMSUNG EX1002, Page 352 of 1031

12. A memory-access method for a memory system with physical memory that includes a plurality of volatile-storage locations addressed by a corresponding plurality of volatile-storage addresses and a plurality of nonvolatile storage locations addressed by a corresponding plurality of nonvolatile-storage addresses, each of the nonvolatile storage locations storing an information unit, the method comprising:

a. copying the information unit from one of the nonvolatile storage locations into one of the volatile storage locations; and

b. saving a mapping associating the volatile-storage address for the one volatile storage location to the nonvolatile-storage address for the one nonvolatile storage location.

13. The method of claim 12, further comprising receiving a write access request for the information unit and writing a modified information unit to the one volatile storage location in response to the request.

14. The method of claim 13, further comprising receiving a second access request for a second one of the information units in a second nonvolatile storage location and, responsive to the second access request, copying the second information unit to the one of the volatile storage locations.

15. The method of claim 14, wherein copying the second information unit to the one of the volatile storage locations includes saving the modified information unit to a previously empty nonvolatile storage location.

16. The method of claim 15, further comprising changing the mapping to associate the

12/21/2022 19:01:46 Page 22 of 31
SAMSUNG EX1002, Page 353 of 1031

volatile-storage address for the one volatile storage location to the previously empty nonvolatile storage location.

17. The method of claim 12, wherein the physical memory further includes a plurality of empty nonvolatile storage locations, the method further comprising erasing some of the information units.

18. The method of claim 17, wherein the erasing occurs when the number of empty nonvolatile storage locations is below a threshold.

19. The method of claim 17, further comprising copying a subset of the information units to be erased to corresponding empty nonvolatile storage locations prior to erasing the information units.

20. The method of claim 19, wherein the subset of the information units to be erased contains valid information units.

21. A system comprising:

a. a memory controller to receive a physical address;

b. memory, coupled to the memory controller, to store an address mapping table accessible by the memory controller; and

c. a plurality of nonvolatile memory ICs coupled to the memory controller, each of the nonvolatile memory ICs having a plurality of nonvolatile access units identified by a respective nonvolatile address;

d. wherein the memory controller uses the mapping table to convert the physical address to one of the nonvolatile addresses.

12/21/2022 19:01:46 Page 23 of 31
SAMSUNG EX1002, Page 354 of 1031

22. The system of claim 21, further comprising a processor coupled to the memory controller to issue an access request to the memory controller specifying the physical address.

23. The system of claim 21, wherein the memory is instantiated in a memory IC separate from the memory controller.

24. The system of claim 21, further comprising a DRAM IC to cache information stored in the nonvolatile memory ICs.

25. The system of claim 21, wherein the DRAM IC includes the memory that stores the address mapping table.

26. A computer-readable medium having stored thereon a data structure defining a memory controller adapted to access a physical memory that includes a plurality of volatile-storage locations addressed by a corresponding plurality of volatile-storage addresses and a plurality of nonvolatile storage locations addressed by a corresponding plurality of nonvolatile-storage addresses, each of the nonvolatile storage locations storing an information unit, the data structure comprising data representing a next-access-unit register to store the nonvolatile-access-unit address of a next one of the nonvolatile access units to receive write data.

27. A memory system comprising:

a. physical memory that includes a plurality of volatile-storage locations addressed by a corresponding plurality of volatile-storage addresses and a plurality of nonvolatile storage locations addressed by a corresponding plurality of

SAMSUNG EX1002, Page 355 of 1031

nonvolatile-storage addresses, each of the nonvolatile storage locations storing an information unit;

b. means for copying the information unit from one of the nonvolatile storage locations into one of the volatile storage locations; and

c. means for saving a mapping associating the volatile-storage address for the one volatile storage location to the nonvolatile-storage address for the one nonvolatile storage location.

28. A memory system comprising:

a. a first nonvolatile memory IC having a first plurality of nonvolatile access units addressed by respective ones of a first plurality of addresses; and

b. a second nonvolatile memory IC having a second plurality of nonvolatile access units addressed by respective ones of a second plurality of addresses;

c. wherein the first nonvolatile memory IC is to store an address table mapping physical addresses to ones of the second plurality of addresses.

29. The memory system of claim 28, further comprising a volatile cache memory to cache the address table in the first nonvolatile memory IC.

12/21/2022 19:01:46 Page 25 of 31
SAMSUNG EX1002, Page 356 of 1031

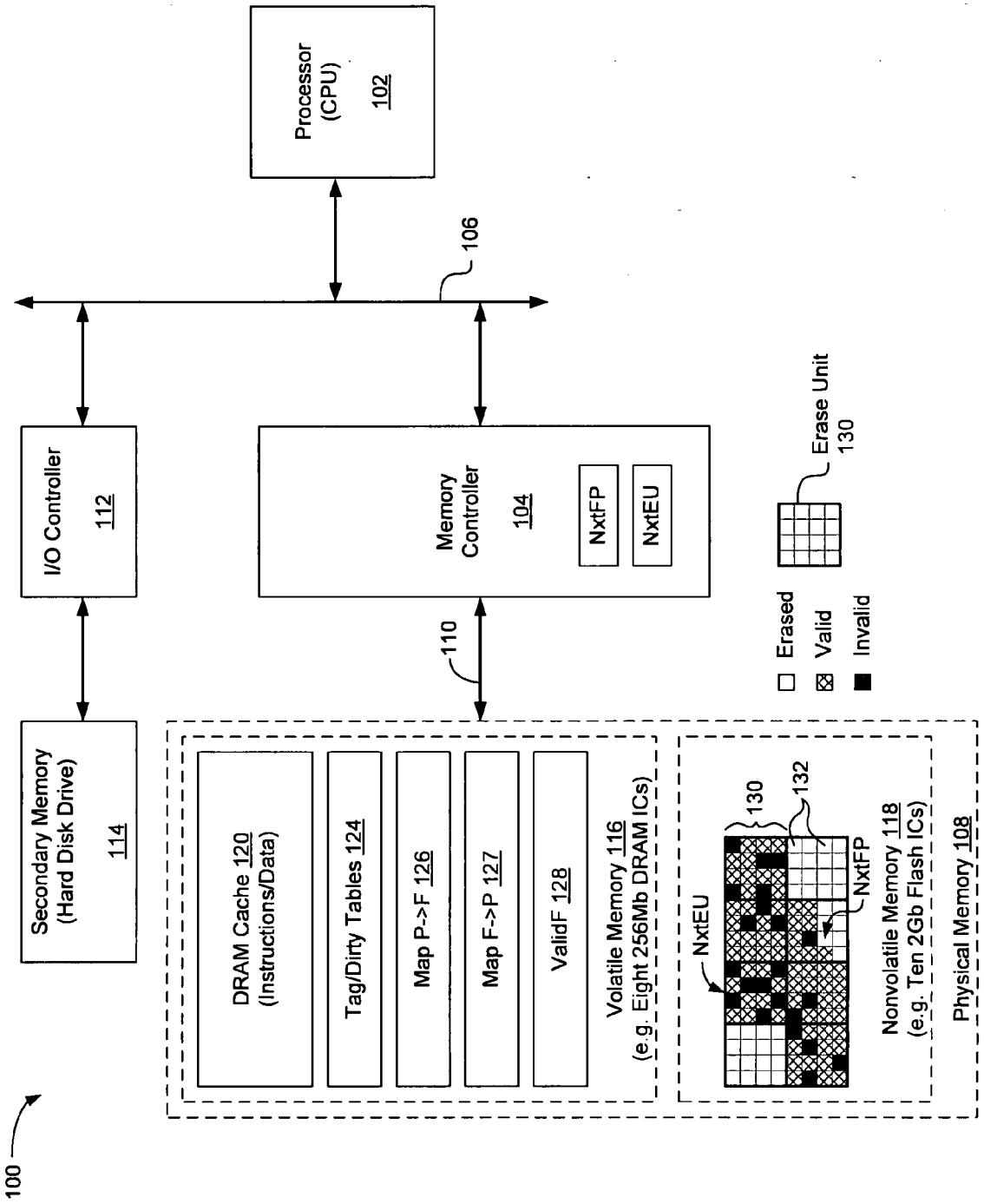

Processor (CPU) 102
106
I/O Controller 112
Memory Controller 104
NxtFP
NxtEU
Erase Unit 130
Erased
Valid
Invalid
110
Secondary Memory (Hard Disk Drive) 114
DRAM Cache 120 (Instructions/Data)
Tag/Dirty Tables 124
Map P->F 126
Map F->P 127
ValidF 128
Volatile Memory 116 (e.g. Eight 256Mb DRAM ICs)
130
132
NxtEU
NxtFP
Nonvolatile Memory 118 (e.g. Ten 2Gb Flash ICs)
Physical Memory 108
100

Fig. 1

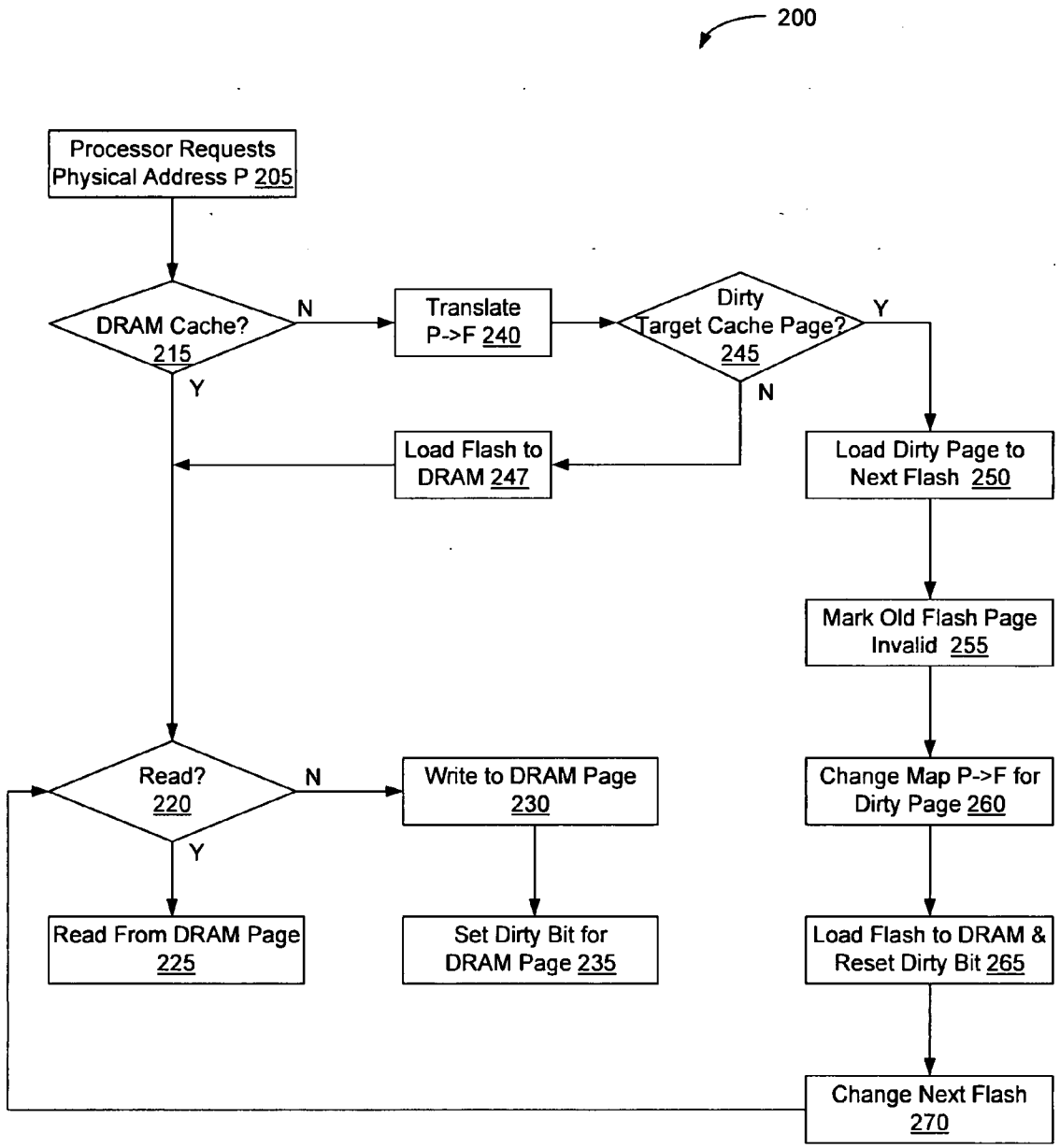


Fig. 2

SAMSUNG EX1002, Page 358 of 1031

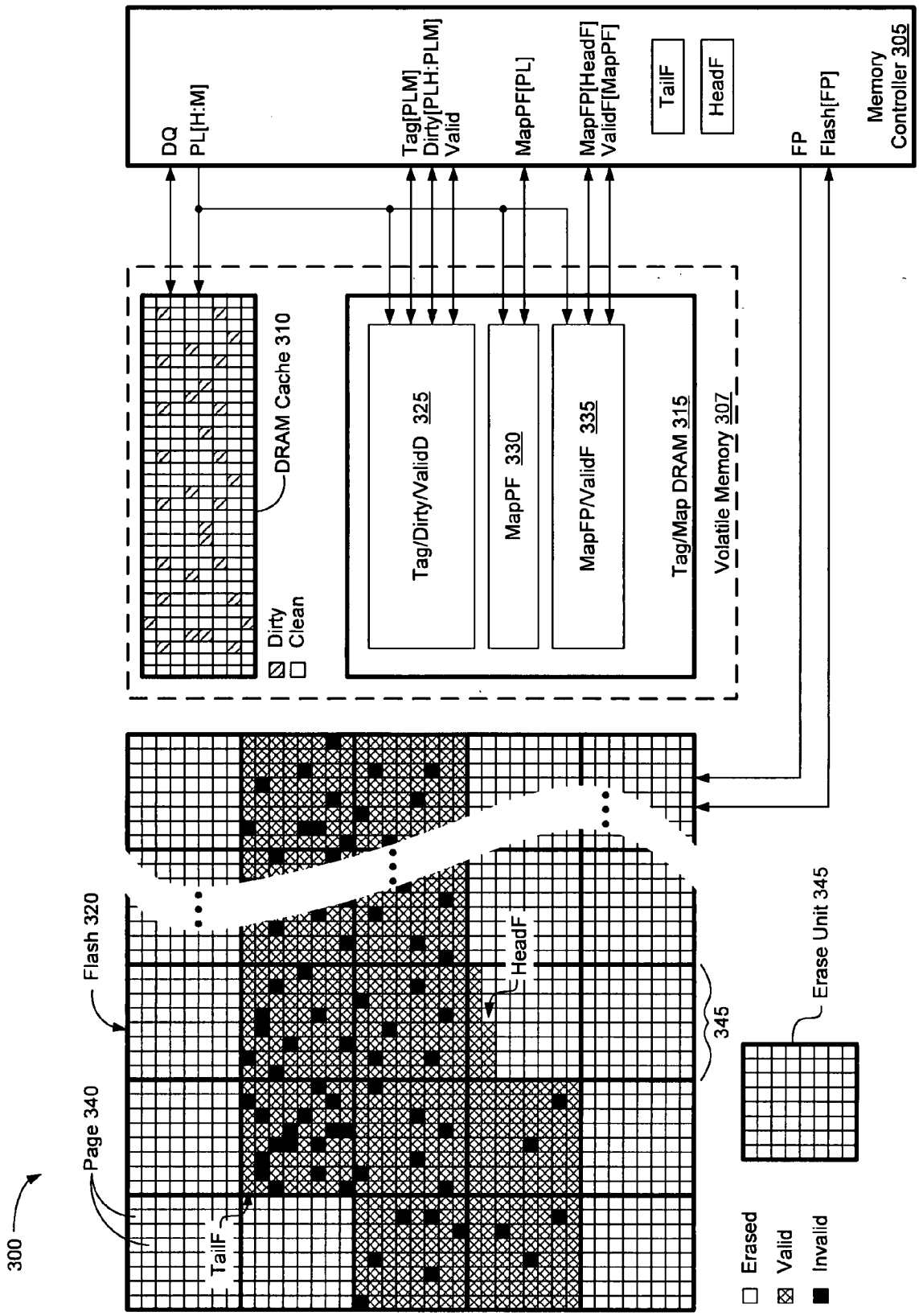

DQ
PL[H:M]
Tag[PLM]
Dirty[PLH:PLM]
Valid
MapPF[PL]
MapFP[HeadF]
ValidF[MapPF]
TailF
HeadF
FP
Flash[FP]
Memory Controller 305
DRAM Cache 310
Dirty
Clean
Tag/Dirty/ValidD 325
MapPF 330
MapFP/ValidF 335
Tag/Map DRAM 315
Volatile Memory 307
Flash 320
HeadF
345
Erase Unit 345
Page 340
TailF
300
Erased
Valid
Invalid

Fig. 3

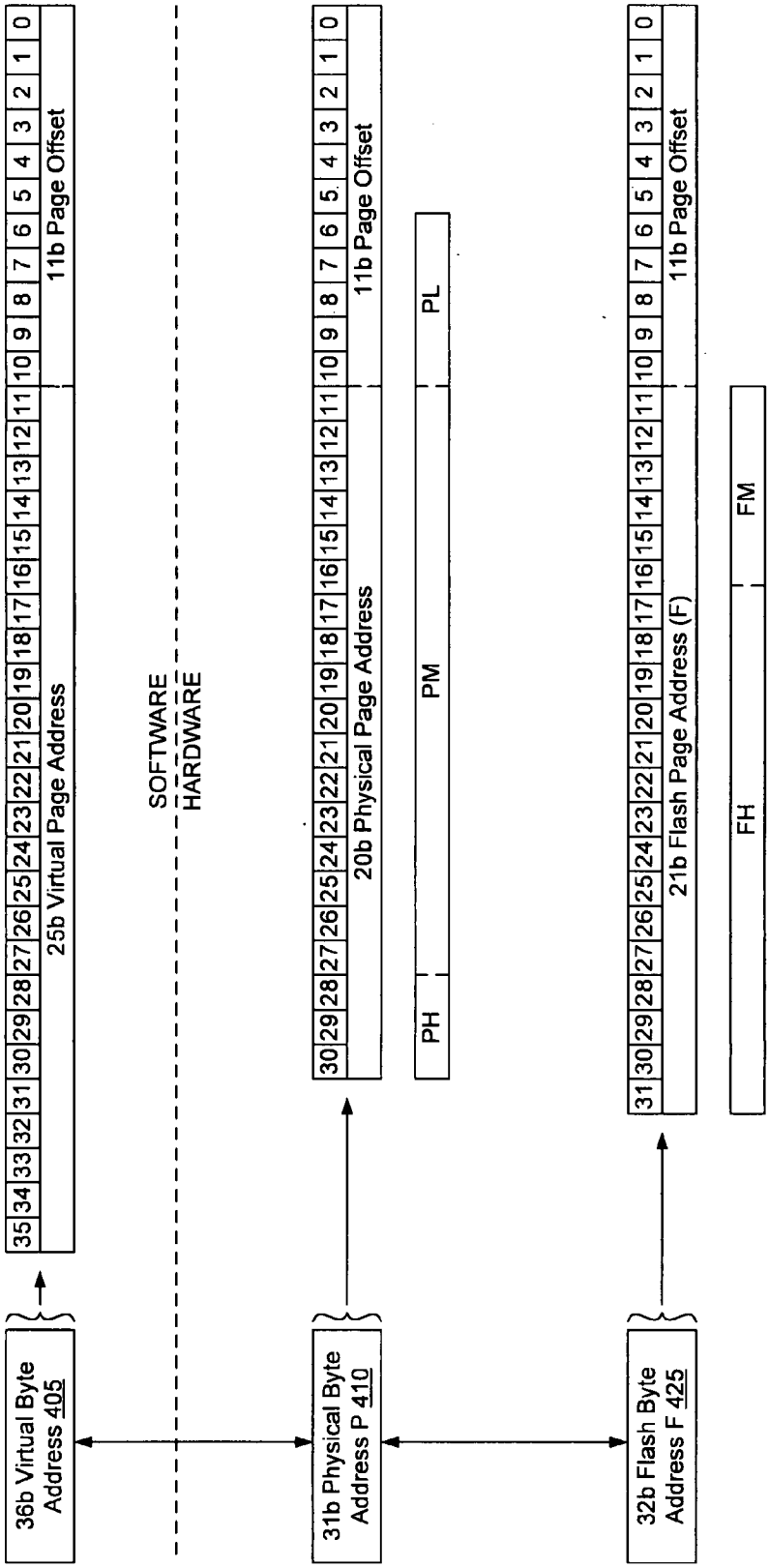


Fig. 4

SAMSUNG EX1002, Page 360 of 1031

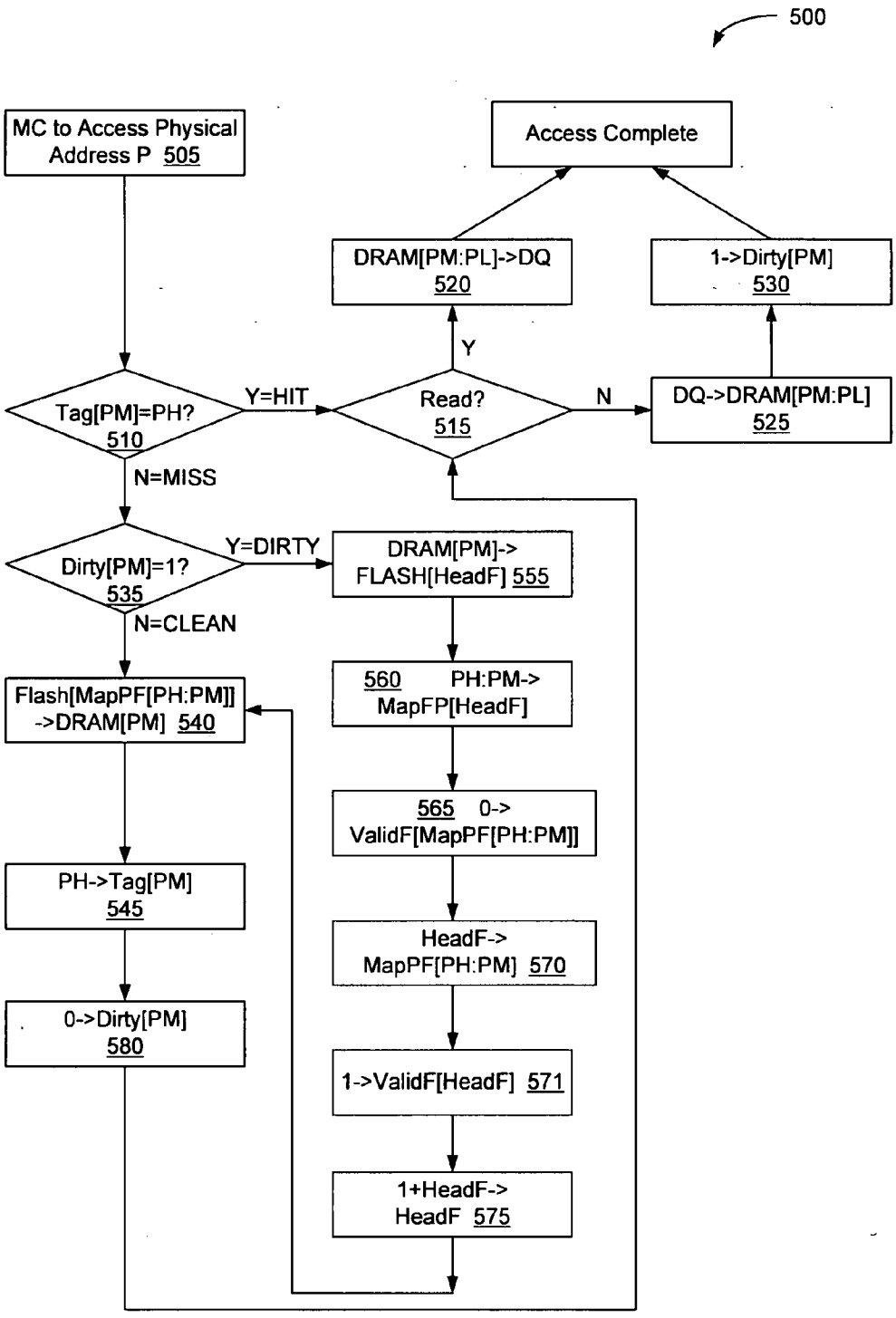


Fig. 5

SAMSUNG EX1002, Page 361 of 1031

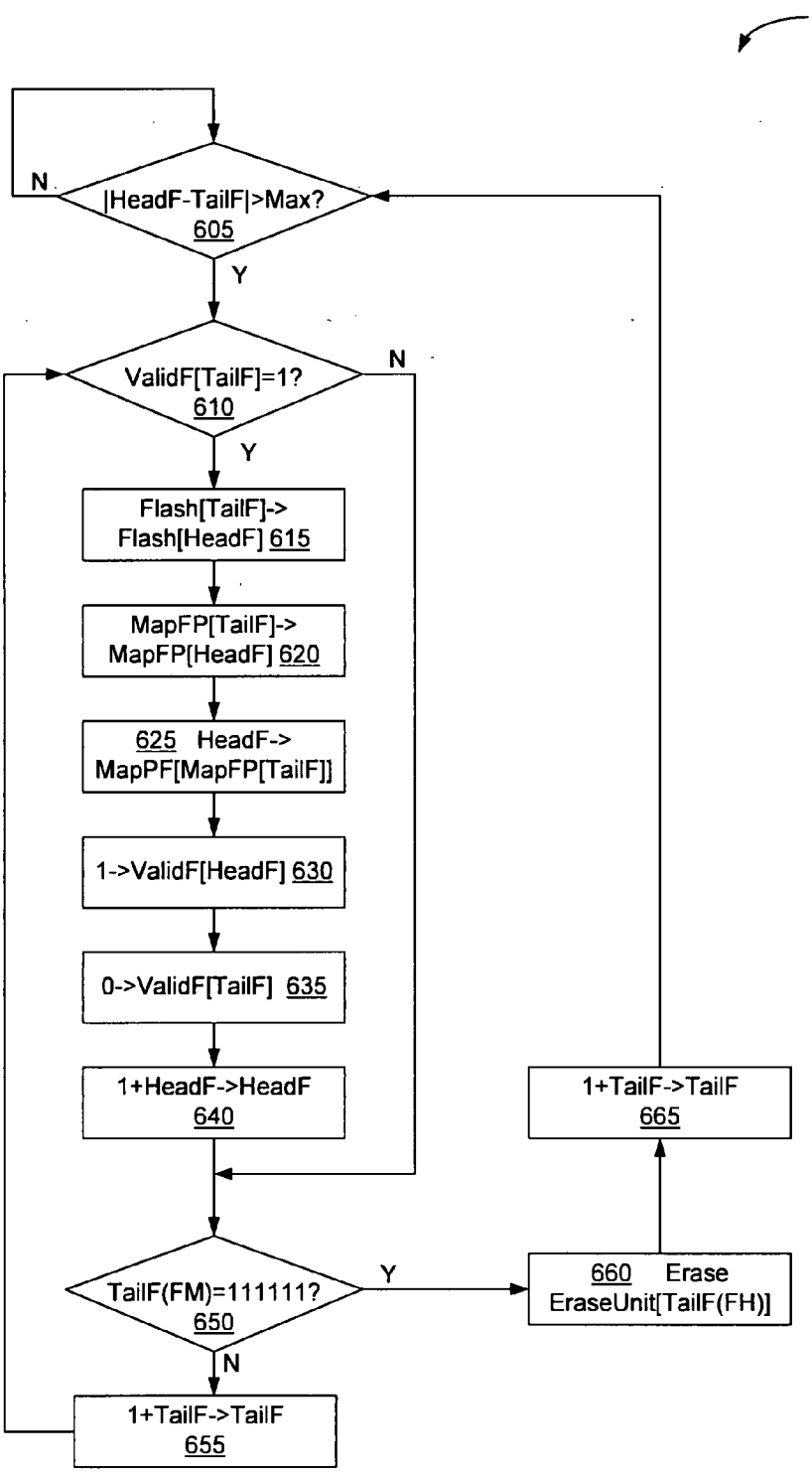


Fig. 6

SAMSUNG EX1002, Page 362 of 1031

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.
17/931,506	09/12/2022	Andrey V. Kuzmin	RMMS.003C12	3894

20995 7590 12/19/2022

KNOBBE MARTENS OLSON & BEAR LLP
2040 MAIN STREET
FOURTEENTH FLOOR
IRVINE, CA 92614

EXAMINER	
BATAILLE, PIERRE MICHE	

ART UNIT	PAPER NUMBER
2136	

NOTIFICATION DATE	DELIVERY MODE
12/19/2022	ELECTRONIC

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

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

efiling@knobbe.com
jayna.cartee@knobbe.com

Applicant-Initiated Interview Summary	**Application No.** 17/931,506	**Applicant(s)** Kuzmin et al.		
	Examiner PIERRE MICHEL BATAILLE	**Art Unit** 2136	**AIA (First Inventor to File) Status** Yes	**Page** 1 of 1

All Participants (applicant, applicants representative, PTO personnel)	**Title**	**Type**
PIERRE MICHEL BATAILLE	Primary Examiner	Telephonic
AARON M DAVIS	Attorney of Record	

Date of Interview: 13 December 2022

Issues Discussed:

Proposed Amendment(s)

Applicant discussed the proposed amendment, proposing: to strike out the phrase "without first erasing" in independent claims 2, 11, and 22; and to rewrite concurrent write operation in claim 1.

Non-statutory Double Patenting

Applicant pledged to file a terminal disclaimer to overcome the double patenting rejection.

/PIERRE MICHEL BATAILLE/ Primary Examiner, Art Unit 2136	

Applicant is reminded that a complete written statement as to the substance of the interview must be made of record in the application file. It is the applicants responsibility to provide the written statement, unless the interview was initiated by the Examiner and the Examiner has indicated that a written summary will be provided. See MPEP 713.04
Please further see:
MPEP 713.04
Title 37 Code of Federal Regulations (CFR) § 1.133 Interviews, paragraph (b)
37 CFR § 1.2 Business to be transacted in writing

Applicant recordation instructions: 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.

Examiner recordation instructions: Examiners must summarize the substance of any interview of record. A complete and proper recordation of the substance of an interview should include the items listed in MPEP 713.04 for complete and proper recordation including the identification of the general thrust of each argument or issue discussed, a general indication of any other pertinent matters discussed regarding patentability and the general results or outcome of the interview, to include an indication as to whether or not agreement was reached on the issues raised.

Doc Code: DIST.E.FILE Document Description: Electronic Terminal Disclaimer - Filed	PTO/SB/25 U.S. Patent and Trademark Office Department of Commerce

Electronic Petition Request	**TERMINAL DISCLAIMER TO OBVIATE A PROVISIONAL DOUBLE PATENTING REJECTION OVER A PENDING "REFERENCE" APPLICATION**
Application Number	17931506
Filing Date	12-Sep-2022
First Named Inventor	Andrey Kuzmin
Attorney Docket Number	RMMS.003C12
Title of Invention	STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

☒ Filing of terminal disclaimer does not obviate requirement for response under 37 CFR 1.111 to outstanding Office Action

☒ This electronic Terminal Disclaimer is not being used for a Joint Research Agreement.

Owner	Percent Interest
Radian Memory Systems, Inc.	100%

The owner(s) of percent interest listed above in the instant application hereby disclaims, except as provided below, the terminal part of the statutory term of any patent granted on the instant application which would extend beyond the expiration date of the full statutory term of any patent granted on pending reference Application Number(s)

17931516 filed on 09/12/2022

as the term of any patent granted on said reference application may be shortened by any terminal disclaimer filed prior to the grant of any patent on the pending reference application. The owner hereby agrees that any patent so granted on the instant application shall be enforceable only for and during such period that it and any patent granted on the reference application are commonly owned. This agreement runs with any patent granted on the instant application and is binding upon the grantee, its successors or 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 of any patent granted on said reference application, "as the term of any patent granted on said reference application may be shortened by any terminal disclaimer filed prior to the grant of any patent on the pending reference application," in the event that any such patent granted on the pending reference application: 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 shortened by any terminal disclaimer filed prior to its grant.

◉ Terminal disclaimer fee under 37 CFR 1.20(d) is included with Electronic Terminal Disclaimer request.

○	I certify, in accordance with 37 CFR 1.4(d)(4), that the terminal disclaimer fee under 37 CFR 1.20(d) required for this terminal disclaimer has already been paid in the above-identified application.

Applicant claims the following fee status:

- (◉) Small Entity
- (○) Micro Entity
- (○) Regular Undiscounted

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

THIS PORTION MUST BE COMPLETED BY THE SIGNATORY OR SIGNATORIES

I certify, in accordance with 37 CFR 1.4(d)(4) that I am:

- (◉) An attorney or agent registered to practice before the Patent and Trademark Office who is of record in this application

  Registration Number 62027

- (○) A sole inventor
- (○) A joint inventor; I certify that I am authorized to sign this submission on behalf of all of the inventors as evidenced by the power of attorney in the application
- (○) A joint inventor; all of whom are signing this request

Signature	/Aaron M. Davis/
Name	Aaron M. Davis

*Statement under 37 CFR 3.73(b) is required if terminal disclaimer is signed by the assignee (owner).
Form PTO/SB/96 may be used for making this certification. See MPEP § 324.

Electronic Patent Application Fee Transmittal

Application Number:	17931506
Filing Date:	12-Sep-2022
Title of Invention:	STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS
First Named Inventor/Applicant Name:	Andrey V. Kuzmin
Filer:	Aaron M. B. Davis/Aimee Kazan
Attorney Docket Number:	RMMS.003C12

Filed as Small Entity

Filing Fees for Utility under 35 USC 111(a)

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
STATUTORY OR TERMINAL DISCLAIMER	2814	1	170	170
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Extension-of-Time:				
Miscellaneous:				
			Total in USD ($)	**170**

Doc Code: DISQ.E.FILE
Document Description: Electronic Terminal Disclaimer – Approved

Application No.: 17931506

Filing Date: 12-Sep-2022

Applicant/Patent under Reexamination: Kuzmin

Electronic Terminal Disclaimer filed on December 13, 2022

☒ APPROVED

This patent is subject to a terminal disclaimer

☐ DISAPPROVED

Approved/Disapproved by: Electronic Terminal Disclaimer automatically approved by EFS-Web

U.S. Patent and Trademark Office

Electronic Acknowledgement Receipt

EFS ID:	47195873
Application Number:	17931506
International Application Number:	
Confirmation Number:	3894
Title of Invention:	STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS
First Named Inventor/Applicant Name:	Andrey V. Kuzmin
Customer Number:	20995
Filer:	Aaron M. B. Davis/Aimee Kazan
Filer Authorized By:	Aaron M. B. Davis
Attorney Docket Number:	RMMS.003C12
Receipt Date:	13-DEC-2022
Filing Date:	12-SEP-2022
Time Stamp:	17:07:12
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	CARD
Payment was successfully received in RAM	$170
RAM confirmation Number	E2022BCH07091713
Deposit Account	111410
Authorized User	Aimee Kazan

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

37 CFR 1.16 (National application filing, search, and examination fees)

37 CFR 1.17 (Patent application and reexamination processing fees)

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Terminal Disclaimer-Filed (Electronic)	eTerminal-Disclaimer.pdf	43892 e4d6f31e704e13e4f8a274ac3ad1c6aaf329d891	no	2
Warnings:					
Information:					
2	Fee Worksheet (SB06)	fee-info.pdf	38283 710147652b10390b4446525538856421419025b8	no	2
Warnings:					
Information:					
		Total Files Size (in bytes):	82175		

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.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventors: Andrey V. Kuzmin, et al.	**Conf. No.:** 3894
App. No.: 17/931,506	**Filed:** September 12, 022
Examiner: Bataille, Pierre Miche	**Art Unit:** 2136
Title: STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS	

AMENDMENT AND REPLY TO OFFICE ACTION

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

Dear Commissioner:

In reply to the Office Action dated November 23, 2022, Applicant respectfully submits the following:

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

Amendments to the Specification begin on page 9 of this paper.

Summary of Interview begins on page 10 of this paper.

Remarks begin on page 11 of this paper.

AMENDMENTS TO THE CLAIMS

Please amend Claims 2-4, 11, 12, 22, 24-26, and 28 as indicated below.

1. (Canceled)

2. (**Currently Amended**) A storage device comprising:

flash memory divided into zones, each zone of the zones being a subdivision comprising a set of erase units, and each erase unit of the set of erase units comprising host accessible pages, wherein each zone:

is associated with a logical address range,

has a zone size for which the storage device is operable to receive a management command from a host that results in the storage device performing an erasure of one or more erase units of the set of erase units,

is in a full state when there are no host accessible pages available for writing ~~without first erasing~~, and

is in an empty state when all host accessible pages are available for writing ~~without first erasing~~; and

logic operable to:

return information about the zones in response to a query command from the host, the information about the zones comprising a list of the zones accessible by the host, a zone state, the zone size, logical addresses associated with respective zones, and information representing an extent to which the respective zones include host accessible pages available for writing ~~without first erasing~~;

~~concurrently~~ write data to the set of erase units of a specified zone using a multi-plane write command such that the data is written concurrently to two or more erase units of the set of erase units; and

update the information representing the extent to which the specified zone includes host accessible pages available for writing ~~without first erasing~~,

wherein the logic comprises at least one of hardware logic or instructions stored on non-transitory machine-readable media that, when executed, control operation of hardware circuitry.

3. (**Currently Amended**) The storage device of Claim 2, wherein when the specified zone has no host accessible pages available for writing ~~without first erasing~~ after ~~concurrently~~ writing data to the set of erase units of the specified zone using the multi-plane write command, the storage device is operable to update a state of the specified zone to the full state.

4. (**Currently Amended**) The storage device of Claim 2, wherein the logic is operable to ~~concurrently~~ write the data to the two or more erase units of the set of erase units of the specified zone using a common page address.

5. (Previously Presented) The storage device of Claim 2, wherein the zone size is a minimum memory size associated with the management command.

6. (Previously Presented) The storage device of Claim 2, wherein the management command from the host results in the logic issuing erase commands to the set of erase units of the specified zone that are on two or more different planes.

7. (Previously Presented) The storage device of Claim 2, wherein the information about the zones is stored on non-volatile memory of the storage device.

8. (Previously Presented) The storage device of Claim 2, wherein the information about the zones is stored on a die that includes the logic.

9. (Previously Presented) The storage device of Claim 2, wherein the information about the zones comprises information indicative of a number of erase operations performed on a particular zone of the zones.

10. (Previously Presented) The storage device of Claim 2, wherein the storage device is operable to:

generate metadata indicating an amount of time since the specified zone was written to, and

in response to comparing the metadata to a threshold, provide an indication that the storage device recommends that the host send a second management command directed to the specified zone.

11. (**Currently Amended**) A storage device comprising:

flash memory divided into zones, each zone of the zones being a subdivision comprising a set of erase units, and each erase unit of the set of erase units comprising host accessible pages, wherein each zone:

is associated with a logical address range,

has a zone size associated with a management command from a host,

transitions to a full state in response to no longer having host accessible pages available for writing ~~without first erasing~~, and

in response to the storage device receiving the management command associated therewith, transitions to an empty state such that all host accessible pages are available for writing ~~without first erasing~~; and

logic operable to:

return information about zones stored by the storage device in response to a query command from the host, the information about the zones comprising:

a list of the zones accessible by the host,

a respective logical address associated with a zone of the zones stored,

the zone size,

information indicating, for respective zones in which there are no host accessible pages available for writing ~~without first~~

~~erasing~~ the respective zone, that the respective zones are in the full state, and

information representing, for respective zones, an extent to which the respective zones include host accessible pages available for writing ~~without first erasing~~;

~~concurrently~~ write data to the set of erase units of a specified zone using a common page address;

update the information representing the extent to which the specified zone includes host accessible pages available for writing ~~without first erasing~~; and

when the specified zone has no host accessible pages available for writing ~~without first erasing~~ the zone after ~~concurrently~~ writing data to the set of erase units of the specified zone, update the information indicating that the specified zone is in the full state,

wherein the logic comprises at least one of hardware logic or instructions stored on non-transitory machine-readable media that, when executed, control operation of hardware circuitry.

12. (**Currently Amended**) The storage device of Claim 11, wherein the logic is operable to ~~concurrently~~ write the data to the set of erase units of the specified zone using a multi-plane write command such that the data is concurrently written to two or more erase units of the set of erase units.

13. (Previously Presented) The storage device of Claim 11, wherein the management command from the host results in the logic issuing at least one erase command to erase one or more erase units of the set of erase units of the specified zone.

14. (Previously Presented) The storage device of Claim 11, wherein the management command from the host results in the logic issuing an erase command associated with erase units of the specified zone that are on two or more different planes.

15. (Previously Presented) The storage device of Claim 14, wherein, as a result of issuing the erase command, the storage device is operable to detect a bad erase unit and mark the bad erase unit as unusable for storage of new write data.

16. (Previously Presented) The storage device of Claim 14, wherein, the storage device is operable to:

generate metadata indicating a count of erasures of the erase units, and

in response to comparing the metadata to a threshold, provide an indication that the storage device recommends that the host send a second management command directed to the specified zone.

17. (Previously Presented) The storage device of Claim 11, further comprising a host interface that comprises buffers to receive the information about the zones and queue the information about the zones for transmission to the host.

18. (Previously Presented) The storage device of Claim 11, wherein the information about the zones is stored on non-volatile memory of the storage device.

19. (Previously Presented) The storage device of Claim 11, wherein the information about the zones is stored on a die that includes the logic.

20. (Previously Presented) The storage device of Claim 11, wherein the information about the zones comprises information indicative of a number of erase operations performed on a particular zone of the zones.

21. (Previously Presented) The storage device of Claim 11, wherein the storage device is operable to:

generate metadata indicating an amount of time since the specified zone was written to, and

in response to comparing the metadata to a threshold, provide an indication that the storage device recommends that the host send a second management command directed to the specified zone.

22. (**Currently Amended**) A storage device comprising:

flash memory divided into zones, each zone of the zones being a subdivision comprising a set of erase units, each erase unit of the set of erase units comprising host accessible pages, and the zones are each associated with a respective logical address range; and

a memory controller in communication with the flash memory, the memory controller comprising:

storage arranged to retain information about the zones, the information about the zones comprising zone size, a logical address associated with a zone of the zones, and information representing an extent to which the zone of the zones comprises host accessible pages available for writing ~~without first erasing~~; and

logic operable to cause the storage device to output the information about the zones in response to a query command from a host, ~~concurrently~~ write to the set of erase units of the zone of the zones using a common page address, and update the information representing the extent to which the zone comprises host accessible pages available for writing ~~without first erasing~~ in response to ~~concurrently~~ writing to the set of erase units of the zone.

23. (Previously Presented) The storage device of Claim 22, wherein the information about the zones comprises a state of the zone.

24. (**Currently Amended**) The storage device of Claim 23, wherein the state of the zone is a full state when the zone has no host accessible pages available for writing ~~without first erasing~~ the zone.

25. (**Currently Amended**) The storage device of Claim 24, wherein the state of the zone is updated to the full state in response to the zone having no host accessible

pages available for writing ~~without first erasing~~ the zone after ~~concurrently~~ writing to the set of erase units of the zone of the zones using the common page address.

26. (**Currently Amended**) The storage device of Claim 23, wherein the state of the zone is an empty state when all host accessible pages of the zone are available for writing ~~without first erasing~~.

27. (Previously Presented) The storage device of Claim 22, wherein the information about the zones comprises information indicative of an amount of time since data was written to the set of erase units of the zone.

28. (**Currently Amended**) The storage device of Claim 22, wherein the information about the zones comprises logical addresses associated with respective ones of a plurality of zones of the zones, and information representing an extent to which each of the plurality of zones comprises host accessible pages available for writing ~~without first erasing~~.

29. (Previously Presented) The storage device of Claim 22, wherein the zone is a minimum memory space associated with a management command from the host.

30. (Previously Presented) The storage device of Claim 22, wherein the logic is operable to cause the storage device to concurrently erase the set of erase units of the zone.

31. (Previously Presented) The storage device of Claim 22, wherein the storage device is operable to:

generate metadata indicating an amount of time since the zone was written to, and

in response to comparing the metadata to a threshold, provide an indication that the storage device recommends that the host send a management command directed to the zone.

AMENDMENTS TO THE SPECIFICATION

Please amend paragraph [0001] as indicated below (insertion underlined):

[0001] This application is a continuation of U.S. Patent Application No. 17/563,481, filed on December 28, 2021 and issued on November 1, 2022 as U.S. Patent No. 11,487,657, which is a continuation of U.S. Patent Application No. 16/751,925, filed on January 24, 2020 and issued on April 26, 2022 as U.S. Patent No. 11,314,636, which is a continuation of U.S. Patent Application No. 16/570,922, filed on September 13, 2019 and issued on November 30, 2021 as U.S. Patent No. 11,188,457, which is a continuation of U.S. Patent Application No. 15/621,888, filed on June 13, 2017 and issued on October 15, 2019 as U.S. Patent No. 10,445,229, which is a continuation of U.S. Patent Application No. 15/346,641, filed on November 8, 2016 and issued on July 18, 2017 as U.S. Patent No. 9,710,377, which is a continuation of U.S. Patent Application No. 14/951,708, filed on November 25, 2015 and issued on December 13, 2016 as U.S. Patent No. 9,519,578, which is a continuation of U.S. Patent Application No. 14/047,193, filed on October 7, 2013 and issued on January 5, 2016 as U.S. Patent No. 9,229,854, which is a continuation in-part of U.S. Patent Application No. 13/767,723, filed on February 14, 2013 and issued on May 16, 2017 as U.S. Patent No. 9,652,376, which claims the benefit of priority to U.S. Provisional Patent Application No. 61/757,464, filed on January 28, 2013. The technical disclosures of each of the aforementioned patent applications are hereby incorporated by reference in their entireties and for all purposes.

SUMMARY OF INTERVIEW

A telephonic interview was conducted on December 13, 2022 and attended by Examiner Pierre Bataille and Applicant's representative, Aaron M. Davis (Reg. No. 62,027). The Terminal Disclaimer being filed herewith and the amendments to independent Claims 2, 11, and 22 in this paper were discussed. Examiner Bataille did not raise any issues with these claim amendments.

REMARKS

This paper is a reply to the Office Action dated November 23, 2022 ("Office Action"). Applicant thanks the Examiner for examining the application and the courtesy extended in the above-summarized Examiner Interview. Paragraph [0001] is amended to recite the patent number of the direct parent application. The language of Claims 2-4, 11, 12, 22, 24-26, and 28 is clarified in this paper. No new matter is being added by these amendments. Claims 2-31 are pending. For at least the reasons presented herein, Applicant respectfully submits that the present application is in condition for allowance.

A. Double Patenting

Claims 2-31 were provisionally rejected on grounds of non-statutory double patenting over Claims 2-25 of co-pending U.S. Patent Application t No. 17/931,516.

Without agreeing that the provisional double patenting rejections are proper or with any assertions in the Office Action, a Terminal Disclaimer over U.S. Patent Application t No. 17/931,516 is being filed herewith. As set forth in M.P.E.P. § 804.02(II), "filing of a terminal disclaimer to obviate a rejection based on nonstatutory double patenting is not an admission of the propriety of the rejection" and "filing of a terminal disclaimer ... raises neither a presumption nor an estoppel on the merits of the rejection." In view of the Terminal Disclaimer, the double patenting rejections are moot. Accordingly, Applicant respectfully requests that Claims 2-31 be allowed.

B. No Disclaimers or Disavowals

Although the present communication may include alterations to the application or claims, Applicant is not conceding in this application that any previously pending claim is not patentable over the art of record. Applicant reserves the right to pursue at a later date any previously pending or other broader or narrower claims that capture any subject matter supported by the present disclosure, including subject matter found to be specifically disclaimed herein. Accordingly, reviewers of this or any parent, child or

related prosecution history shall not reasonably infer that Applicant has made any disclaimers or disavowals of any subject matter supported by the present application.

C. Conclusion

In view of the above, Applicant respectfully submits that the application is in condition for allowance. If any questions remain, the Examiner is invited to contact the undersigned attorney at the number provided below. Please charge any additional fees, including any fees for additional extension of time, or credit overpayment to Deposit Account No. 11-1410.

Respectfully submitted,

KNOBBE, MARTENS, OLSON & BEAR, LLP

Dated: December 13, 2022

By: /Aaron M. Davis/
Aaron M. Davis
Registration No. 62,027
Attorney of Record
(858) 707-4000

56743722

JP2001051889A FILE SYSTEM USING NONVOLATILE SEMICONDUCTOR STORAGE DEVICE

Data originating from sources other than the EPO may not be accurate, complete, or up to date.

The wording below is an initial machine translation of the original publication. To generate a version using the latest translation technology, go to the original language text and use Patent Translate.

BACKGROUND OF THE INVENTION 1. Field of the Invention The present invention relates to a file system using a nonvolatile semiconductor memory device such as a flash memory.

2. Description of the Related Art Recently, a nonvolatile semiconductor memory device (for example, a flash memory) capable of electrically writing and erasing data has a hard disk drive capable of magnetically writing and erasing data. The technique for using it as a substitute for is attracting attention.

The flash memory has a data read speed of about 10,000 times and a data write speed of about 1,000 times as compared with a hard disk drive, and it is easy to reduce the size and weight, and is excellent in impact resistance. It has features such as high portability, and is suitable for the purpose of storing and reading low-capacity data at high speed.

However, in the flash memory, in order to write new data to a storage area in which data has already been written, it is necessary to once erase the written data. Moreover, the unit for erasing data is as large as several kilobytes, a relatively long data erasing time of tens of milliseconds to several seconds is required, and there is a drawback that the guaranteed number of times of writing and erasing data is finite.

Therefore, when a flash memory is used as a substitute for a hard disk drive, in view of the above-mentioned drawbacks, the unit of data handling is reduced to a few hundred bytes, which is similar to that of a hard disk drive, and data erasing is not performed as much as possible, so that data can be efficiently stored. It is necessary to provide a file management system for flash memory, that is, a file system that supports flash memory. Such a file system is disclosed in, for example, Japanese Patent Application Laid-Open No. 6-95955.

Further, US Pat. No. 5,544,119 shows a processing method at the time of power restoration after power supply interruption, and US Pat. No. 5,544,356 shows a method for reconstructing a file system. Has been done.

However, in the technique described in the above publication, a file system that performs flash memory and various controls in an environment in which power supply using a battery or the like as a power source is unstable When the flash memory or file system is in operation and the power supply is interrupted for a moment due to battery exhaustion, that is, when the power supply is cut off unintentionally, There was a possibility that the data stored in the memory could not be read normally.

Specifically, firstly, the present state of a storage area (hereinafter referred to as a sector) in which data is stored is represented by sector state information. This sector state information is "unused", Since it only represents one of the three states of "in use" and "invalid", if the power supply is cut off when updating the sector data, the data may not be updated normally. Sometimes, data was regarded as normal based on the sector status information. The data update processing of this sector will be described below with reference to the flowchart of FIG.

According to the flow chart of FIG. 10, first, the sector status information of one sector storing the existing data to be updated is changed from "in use" to "invalid" (step 101), and then another. The sector state information of one sector is changed from "unused" to "in use" (step 102), and then the update data is written to the sector whose sector state information becomes "in use" (step 103). . Steps 101 to 103 are repeated until there is no data to write.

Here, immediately after step 102 or during execution of step 103, if the power supply is cut off and the writing of the data to the flash memory is interrupted, the writing of the update data to the sector is not normally completed. Nevertheless, the sector status information of this sector is "in use". Therefore, when the power supply is restored, the file system mistakenly recognizes the data stored in this sector as valid.

Second, when power is cut off when erasing all the data in one block and rebuilding the file system, the same data is stored in two blocks, or in one block. Occasionally, a state in which all the data in the above had not been

completely erased. The process of rebuilding the file system will be described below with reference to the flowchart of FIG.

According to the flowchart of FIG. 11, first, all the sectors are erased, and a spare block in which the sector state information of all the sectors is "unused" is reserved in advance, and all the data is erased. Data to be stored, that is, the data of the sector whose sector status information is "invalid" and all other data except the sector status information are copied to the spare block (step 111), and all the data is erased. The block number included in the position information other than the sector of which the sector status information in is "invalid" is updated to the block number of the spare block (step 112). Finally, the block from which the data is copied is erased and secured as a spare block for the next file system reconstruction (step 113).

If power supply is interrupted before execution of step 113 or during execution of step 113, when the power supply is restored, the same data is stored in two blocks, or one block is stored. A state occurs in which all the data in the above cannot be completely erased, and a new spare block cannot be secured.

Therefore, the present invention has been made in view of the above-mentioned conventional problems, and a nonvolatile semiconductor capable of normally reading the data stored in the flash memory even when the power supply is cut off. An object is to provide a file system using a storage device.

In order to solve the above problems, a file system of the present invention is a file system using a nonvolatile semiconductor memory device having a plurality of erase blocks each including a plurality of sectors. A storage unit that stores, for each erase block, any one of a plurality of state information about the erase block, and stores, for each sector, any one of a plurality of state information about the sector; When the semiconductor memory device is accessed, a guarantee means for guaranteeing the data already stored in the nonvolatile semiconductor memory device based on the state information of each erase block and the state information of each sector in the memory means. To have.

According to the present invention, a plurality of status information about blocks and a plurality of status information about sectors are stored in the storage means. Therefore, even if the power supply is unintentionally cut off while the file system is operating, when the power supply is restored, a plurality of status information about blocks and a plurality of statuses about sectors at the time of initialization processing at startup. Based on the information, it is possible to accurately grasp the state of blocks and sectors and manage the flash memory in detail, without destroying the existing data stored in the flash memory and without causing inconsistency in the existing data. It is possible to automatically restore the file system to the previous stage that was being executed immediately before the power supply was cut off, without causing it.

In one embodiment, the storage means is provided on a RAM that has no restrictions on data updates, and the guarantee means is provided for each erase block on the RAM when the nonvolatile semiconductor storage device is accessed. The status information of each sector and the status information of each sector are referred to.

By using the RAM in this way, there is no restriction on the updating of data and the processing speed is increased.

In one embodiment, at least five states of "unused", "data present", "block full", "data transferring", and "source block erasing" are used as a plurality of status information about the erase block. A state is set, and the assurance means updates the state information of the spare block previously reserved for reconstruction from "unused" to "data being transferred", and the erase block in the data transfer source. Means for transferring and storing the data of the spare block to the spare block, and after the transfer and storage of the data are completed, the block status information of the spare block is updated from "data being transferred" to "erasing the original block". The data in the erase block of the data transfer source is erased, the erase block of the data transfer source is secured as a new spare block, and the status information of the new spare block is changed from "with data" or "block full" to "unused". And a means for updating the status information of the spare block of the data transfer destination from "erasing the original block" to "with data".

In this way, when the five states of "unused", "data present", "block full", "data being transferred", and "source block being erased" are used as the erase block state information, It is possible to accurately grasp the status and manage the flash memory in detail, and cut off the power supply without destroying the existing data stored in the flash memory and without causing inconsistency in the existing data. It is possible to automatically restore the file system to the previous stage that was being executed immediately before.

In one embodiment, at least one spare block reserved in advance for reconstruction is provided for each erase block having a different block size included in one nonvolatile semiconductor memory device.

In this case, there is an advantage that it is possible to employ a flash memory having a uniform block or boot block structure, which is suitable for the purpose of the system or which is cheaper.

In one embodiment, at least one spare block previously reserved for reconstruction is provided for each erase block having a different block size included in a plurality of nonvolatile semiconductor memory devices.

In this case, there is an advantage that the number of spare blocks required by the file system can be reduced. Therefore, there is also an advantage that the file system can be constructed inexpensively.

In one embodiment, the spare block previously reserved for reconstruction has a size equal to or greater than the size of the erase block.

In this case, as a spare block for the boot block (for example, a size of 8 Kbytes), a block having a size larger than 8 Kbytes (for example, a size of 64 Kbytes) can be allocated to the spare block to make the size of the spare block constant. Therefore, management becomes easy.

The file system of the present invention is, in a file system using a nonvolatile semiconductor memory device having a plurality of sectors, storage means for storing, for each sector, any one of a plurality of state information about the sector. A sector for guaranteeing data already stored in the nonvolatile semiconductor memory device based on the state information of each sector in the memory device when the nonvolatile semiconductor memory device is accessed. At least five states of "unused", "data writing", "data writing completed", "data valid", and "data invalid" are set as a plurality of state information about A means for updating the status information of the sector to which the data is written from "unused" to "data writing", and the status information of the sector from "data writing" to "data writing" after the data writing to the sector is completed. When data is newly written to the sector, the state information of the sector is updated from "data write completed" to "data valid", and the existing data in the sector is updated. When data is written, the status information of the other sector in which the existing data is written is updated from “data valid” to “data invalid”, and then the status information of the sector in which the updated data is written is updated. Update from "Data write completed" to "Data valid".

According to the present invention, five status information about the sector is stored in the storage means. Therefore, even if the power supply is unintentionally cut off while the file system is operating, when the power supply is restored, the status of the sector can be determined based on multiple status information about the sector during the initialization process at startup. Just before shutting down the power supply, it is possible to accurately manage the flash memory and manage the flash memory in detail, without destroying the existing data stored in the flash memory and without causing inconsistency in the existing data. It is possible to automatically recover the file system to the previous stage that was being executed.

In another embodiment, the plurality of status information about the erase block is "unused", "data present", "data being transferred", "source block is being erased", and "source block is being formatted". At least five states may be set.

In this case, means for updating the state information of the spare erase block previously reserved for reconstruction from "unused" to "data being transferred", and the data in the erase block of the data transfer source are stored in the spare block. Means for transferring and storing to the erase block, and after completion of transfer and storage of the data, update the block state information of the preliminary erase block from "during data transfer" to "during original block erase", and Means for erasing the data in the erase block, securing the erase block of the data transfer source as a new preliminary erase block, and updating the status information of the new preliminary erase block from "with data" to "unused"; The state information of the preliminary erase block of the data transfer destination is updated from "source block erasing" to "source block formatting", and the erasable count of the new preliminary erase block is written to the new preliminary erase block of the data transfer source. A configuration including (formatting) means and means for updating the state information of the preliminary erase block of the data transfer destination from "during original block formatting" to "with data" after the formatting is completed.

In this way, when the five states of "unused", "data present", "data being transferred", "source block being erased", and "source block being formatted" are used as the erase block state information. , It is possible to accurately grasp the state of the erase block and manage the flash memory in detail, without destroying the existing data stored in the flash memory and without causing inconsistency in the existing data, It is possible to automatically restore the file system to the previous stage that was being executed immediately before the power supply was cut off. Further, the method of not setting the state of “block full” as the state information of the erase block is particularly effective when the erase block is divided into sectors of uneven sizes and used. This is because when writing data of unequal size to consecutive sectors in the same erase block, if the size of the data to be written next is unknown, the state information of the erase block becomes “block full”. This is because it is difficult to determine whether or not it is.

In the initialization of the file system when the power supply is restored, it is searched whether or not there is an erasing block "erasing the original block" as the state information of the erasing block, and if the erasing block exists, An erase block having the same logical block number as the erase block and the erase block status information being “with data” or “block full” is selected from the erase block status information in the storage means. A means for acquiring, a means for erasing the acquired erase block data, and a method for updating the state information of this erase block from "with data" or "block full" to "unused" after the erase is completed, and the state of the erase block It may include means for updating the state information of the erase block whose information is “erasing the original block” from “erasing the original block” to “with data”.

At least five states of "unused", "data present", "data being transferred", "source block being erased", and "source block being formatted" are set as a plurality of state information about the erase block. The assurance means updates the status information of the preliminary erase block reserved in advance for the reconstruction of the file system from "unused" to "data being transferred", and the erase block of the data transfer source. Means for transferring and storing the data in the pre-erase block to the pre-erase block, and updating the state information of the pre-erase block from "data being transferred" to "source block being erased" after completion of transfer and storage of the data, After erasing the data in the erase block of the data transfer source, after the erase is completed, the erase block of the data transfer source is secured as a new spare erase block, and the state information of this new spare erase block is changed from "data present" to "not present". And a means for updating the status information of the preliminary erase block of the data transfer destination from "erasing the original block" to "formatting the original block" and formatting a new preliminary erase block of the data transfer source. And a means for updating the status information of the preliminary erase block of the data transfer destination from "during original block formatting" to "with data" after completion of formatting.

The erase block stores a block erasable number of times in each erase block, and at the time of the formatting, a new preliminary erase block is subtracted from the erasable number of the preliminary erase block by one. A means for storing the data in the erase block and a means for selecting the erase block having the largest block erase count as a data transfer source among all the erase blocks except the preliminary erase block when the file system is reconstructed But it's okay.

The use of the erase block may be prohibited when the block erasable count reaches a value of 0 or less.

In the erase block, the number of blocks that can be erased is stored in each erase block, and the number of times the erase count of the spare erase block is incremented by 1 is added to the new erase block during the formatting. Even if the means for storing in the block and the means for selecting the erase block having the largest block erase count among all the erase blocks excluding the preliminary erase block as the data transfer source at the time of rebuilding the file system, Good.

The use of the erase block may be prohibited when the block erasable count becomes equal to or larger than the specification value (maximum value) of the erasable count.

Preliminary erase blocks reserved in advance for file system reconstruction may have at least one erase block for each erase block having a different block size included in one nonvolatile semiconductor memory device. ..

Preliminary erase blocks reserved in advance for reconstructing the file system may have at least one erase block for each erase block having a different block size included in the plurality of nonvolatile semiconductor memory devices. ..

The preliminary erase block previously reserved for the reconstruction of the file system may have a size equal to or larger than the erase block size.

In a file system using a non-volatile semiconductor memory device having a plurality of sectors, a storage means for storing, for each sector, any one of a plurality of status information about the sector, and the non-volatile semiconductor memory device. At the time of access, guarantee means for guaranteeing the data already stored in the nonvolatile semiconductor memory device based on the status information of each sector in the storage means, Is set to at least five states of "unused", "data writing", "data writing completed", "data valid", and "data invalid". A means for updating the status information from "unused" to "data writing", and after the data writing to the sector is completed, the status information of the sector is updated from "data writing" to "data writing completed". Means and when data is newly written in the sector, the state information of the sector is updated from “data write completion” to “data valid”, and when the updated data of existing data is written in the sector, After updating the status information of the other sector in which the existing data is written from "data valid" to "data invalid", the status information of the sector in which the updated data is written is changed from "data write completion" to "data write complete". Means for updating to "data valid".

In the initialization of the file system when the power supply is restored, it is searched whether or not there is an erasing block of "original block erasing" as the state information of the erasing block. If the erasing block is present, it is searched. A means for acquiring the data transfer source block to the erase block from the status information of each erase block in the storage means and the acquired data of the data transfer source block are erased, and after the erase is completed, the status information of this erase block is A means to update from "with data" to "unused" and the status information of the erase block whose erase block status information is "erasing the original block" from "erasing the original block" to "formatting the original block", A means to format the erased block that erased the data and secure it as a new preliminary erased block, and the state information of the erased block whose original state is "Original block formatting" after the completion of formatting, To “with data” may be included.

In the initialization of the file system when the power supply is restored, it is searched whether or not there is an erase block of "original block format" as the state information of the erase block. Means for acquiring the data transfer source block to the erase block from the status information of each erase block in the storage means, means for formatting the acquired data transfer source block and securing it as a new preliminary erase block, and after the completion of formatting A unit for updating the state information of the erase block whose state information of the erase block is “in the original block format” from “in the original block format” to “with data” may be included.

Embodiments of the present invention will be described below with reference to the accompanying drawings.

FIG. 1 shows an embodiment of the file system of the present invention. In FIG. 1, 1 is a file system, 2 is a file system control unit that controls processing of the file system 1, 3 is a file system memory unit used by the file system 1 to store control information, and 4 is a flash memory unit 5. Is a flash memory control unit for controlling data processing to the file system 1, 5 is a flash memory unit, and 6 is an application or operating system for requesting the file system 1 to process data.

FIG. 2 is a configuration diagram of data stored in the flash memory unit 5 in FIG. Each erase block 51 of the flash memory unit 5 is divided into a plurality of physical sectors 52, which are units for handling data, and arranged, and each erase block 51 has a unique physical sector number. If the size of each erase block 51 is 64 Kbytes and the size of each physical sector 52 is 512 bytes, there will be 128 physical sectors 52 in each erase block 51.

Here, an arbitrary one erase block 51 among the plurality of erase blocks 51 in the flash memory unit 5 is used as a data-unwritten preliminary erase block used for reconstruction of the file system 1, and each erase block 51 is also used. The sector 52 at the head of the erase block 51 is reserved in advance as a block control sector 52a for storing control information of each erase block.

In the erase block 51, the block control sector 52a has 2 bytes as the state information of the erase block 51 from the beginning, 2 bytes as the state information of the physical sector 52 of the sector #1, and sectors #2 to #127. Each 2 bytes as the status information of each physical sector 52 are sequentially arranged, and thus have information of 2 bytes×128=256 bytes.

The state information of the erase block 51 is composed of logical block numbers 0 to 255 assigned by the file system control unit 2 and data indicating one of five types of block states. The five types of block states are: (1) 11111111b (b is a binary number; the same applies hereinafter) representing "unused", (2) 11111110b

representing "data being transferred", and (3) "source block is being erased". 11111100b representing "," 11111000b representing (4) "data present", and 11110000b representing (5) "block full".

The state information of each sector 52 of the erase block 51 is, like the state information of the erase block 51, a logical sector number unique in the system of 0 to 4095 to which the file system control unit 2 allocates, and five kinds. It is composed of data representing any one of the sector states. The five types of sector states are (I) 1111b indicating "unused", (II) 1110b indicating "data writing", (III) 1100b indicating "data writing completed", and (IV) "data valid". Is 1000b, and (V) is 0000b indicating "data invalid".

FIG. 3 is a data structure diagram (hereinafter referred to as block information table 10) for each erase block 51 stored in the file system memory unit 3 in FIG.

In FIG. 3, a physical block number, a logical block number, and a block state are one information unit. For example, a physical block number=0, a logical block number=0, and a block state=data exists. 51, the physical block number=1, the logical block number=1, and the erase state 51 in which the block state=block full is present.

FIG. 4 is a configuration diagram (hereinafter referred to as sector information table 11) of data regarding each sector 52 stored in the file system memory unit 3 in FIG.

In FIG. 4, a physical block number, a physical sector number, a logical sector number, and a sector state are one information unit. For example, physical block number=0, physical sector number=0, logical sector number=100, sector It is shown that there are a sector 52 in which status=data is valid, a physical block number=0, a physical sector number=1, a logical sector number=10, and a sector 52 in which sector status=data is invalid.

FIG. 5 is a block diagram of data in which the number of each sector 52 is arranged for each erase block 51 stored in the file system memory unit 3 in FIG. It is referred to as the information table 12).

In FIG. 5, the physical block number, the number of unused sectors, the number of data valid sectors, and the number of data invalid sectors are one information unit. For example, physical block number=0, number of unused sectors=100, data valid. Erase block 51 with sector number=20, data invalid sector number=7, physical block number=1, unused sector number=0, data valid sector number=50, data invalid sector number=77 This indicates that the block 51 and the like exist.

The block information table 10 of FIG. 3, the sector information table 11 of FIG. 4, and the sector information table 12 of FIG. 5 are block control of each erase block 51 from the flash memory section 5 at the stage when the file system 1 is activated. It is created by reading the sector 52a.

Even when the block information table 10, the sector information table 11 and the sector information table 12 are not created, the information of these tables is stored in the block control sector 52a of each erase block 51, so that the file system 1 It is possible to use However, since the operation speed of the file system memory unit 3 is faster than that of the flash memory unit 5, it is necessary to create and store the block information table 10, the sector information table 11, and the sector information table 12 in the file system memory unit 3. However, the internal state of the flash memory unit 5 and the data storage location can be grasped at high speed. Further, since the data exchange with the flash memory unit 5 is reduced, it is very effective to use the file system memory unit 3 for increasing the operating speed of the file system 1.

FIG. 6 is a flow chart relating to the process of reading the data stored in the flash memory unit 5 in this embodiment.

First, when a request for reading data from the flash memory unit 5 is issued in the application or operating system 6, a logical sector number is given from the application or operating system 6 to the file system control unit 2 together with this read request. The file system control unit 2 accesses the sector information table 11 in the file system memory unit 3 to search for the given logical sector number, and the sector state corresponding to this logical sector number becomes "data valid". It is confirmed that the physical sector number corresponding to this logical sector number is obtained, and this physical sector number is given to the flash memory controller 4 (step 121). The flash memory control unit 4 accesses the flash memory unit 5 based on this physical sector number, reads the sector of the physical storage location corresponding to this physical sector number, and controls the read data to the file system. It is given to the part 2 (step 122). If there is the remaining

data to be read, the process returns to step 121 (step 123, Yes), and if there is the remaining data to be read, the process ends (step 123, No).

FIG. 7A to FIG. 7C are flowcharts relating to the process of writing data to the flash memory unit 5 in this embodiment.

First, when an application or operating system 6 issues a data write request to the flash memory unit 5, the logical sector number and the data to be written are given from the application or operating system 6 to the file system control unit 2 together with this write request. Be done. The file system control unit 2 accesses the sector information table 12 in the file system memory unit 2 and acquires the physical block number (indicating the writable erase block 51) having the largest number of unused sectors (step 131). If the physical block number cannot be acquired (step 132, No), the file system 1 described later is re-established to secure the writable erase block 51. Construction is performed (step 133).

If the physical block number having the largest number of unused sectors can be obtained (step 132, Yes), the file system control unit 2 accesses the sector information table 11 to obtain the physical block obtained in step 131 or 133. The number of unused sectors corresponding to the physical block number is obtained by obtaining the physical sector number whose sector status is “unused” from among the physical sector numbers corresponding to the number, and further accessing the sector information table 12. Is set to -1 and the number of valid data sectors is set to +1 (step 134).

Thereafter, the file system controller 2 accesses the block information table 10 and confirms whether the block state corresponding to the physical block number acquired in step 131 or 133 is "unused" (If the block state is "unused" (step 135) (Yes at step 135), the flash memory unit 5 is accessed through the flash memory control unit 4 and the block control sector 52a of the erase block 51 corresponding to the physical block number is accessed. To the physical block number, and the block state of the block control sector 52a is updated from "unused" to "data present". Similarly, the block information table 10 is accessed, the logical block number corresponding to the physical block number acquired in step 131 or 133 is updated to the physical block number, and the block state is updated from "unused" to "data available". (Step 136), the process moves to step 137. If the block state is not "unused" (step 135, No), step 136 is jumped to step 137.

Next, the file system controller 2 accesses the sector information table 12, and confirms whether the number of unused sectors corresponding to the physical block number acquired in step 131 or 133 is 0 (step 137), If the number of unused sectors is 0 (step 137, Yes), the flash memory unit 5 is accessed through the flash memory control unit 4, and the block state of the block control sector 52a of the erase block 51 corresponding to the physical block number is set to " Update from "with data" to "block full". Similarly, the block information table 10 is accessed, and the block status corresponding to the physical block number acquired in step 131 or 133 is also updated from “with data” to “block full” (step 138).

Next, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4, and erases the physical block number acquired in step 131 or 133 and the physical sector number acquired in step 134. The logical sector number given to the file system controller 2 is written in the logical sector number of the block control sector 52a of the block 51, and the sector state of this sector 52 is updated from "unused" to "data writing". Similarly, the sector information table 11 is accessed, and the physical block number acquired in step 131 or 133 and the logical sector number corresponding to the physical sector number acquired in step 134 are set to the logical sector number given to the file system control unit 2. The sector status is updated from "unused" to "data being written" (step 139).

After that, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4, and the sector corresponding to the physical block number acquired in Step 131 or 133 and the physical sector number acquired in Step 134. The data given to the file system controller 2 is written in 52 (step 140).

After the completion of writing to the sector 52 in the flash memory unit 5, the file system control unit 2 writes the erase block 51 corresponding to the physical block number acquired in Step 131 or 133 and the physical sector number acquired in Step 134. The sector state of the block control sector 52a is updated from "data writing" to "data writing completed". Similarly, the sector information table 11 is accessed and the sector state corresponding to the physical block number acquired in step 131 or 133 and the physical sector number acquired in step 134 is updated from "data writing" to "data writing completed". (Step 141).

Then, the file system control section 2 confirms whether the data written in the sector 52 in the flash memory section 5 is the update data of the data already written in the flash memory section 5 (step 142). . Here, if the existing data is updated (step 142, Yes), the file system control unit 2 accesses the sector information table 11 and has the same logical sector number as the logical sector number of the sector 52 to which the writing is performed in step 140. Physical sector number and physical block number of another sector 52 that has a number and is different from the physical sector number of the sector 52 that has been written in step 140, that is, the physical sector number and physical block of the sector 52 in which existing data is stored. The block number is acquired (step 143). Then, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4, and sets the sector state of the block control sector 52a of the erase block 51 corresponding to these physical block numbers and physical sector numbers to "data valid". Update from" to "Data invalid". Similarly, the sector information table 11 is accessed to update the physical block number and the sector state corresponding to the physical sector number from "data valid" to "data invalid". Further, the sector information table 12 is accessed to set the number of valid data sectors corresponding to this physical block number to -1 and the number of invalid data sectors to +1 (step 144).

After that, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4 and erases the physical block number acquired in step 131 or 133 and the physical sector number acquired in step 134. The sector state of the block control sector 52a of the block 51 is updated from "data write completion" to "data valid". Similarly, the sector information table 11 is accessed to update the physical block number and the sector state corresponding to the physical sector number from "data write completion" to "data valid" (step 145).

Subsequently, the file system controller 2 confirms whether the write data still remains (step 146). If the write data still remains (step 146, Yes), the procedure returns to step 131, When there is no write data (step 146, No), the process ends.

8A and 8B are flowcharts relating to the rebuilding process of the file system 1 in this embodiment.

First, when the writable erase block 51 cannot be acquired as described in the flowcharts of FIGS. 7A to 7C (step 132, No), the file system is used to secure the writable erase block 51. Is reconstructed (step 133). The process of step 133 corresponds to the process shown in the flowchart of FIG.

8A and 8B, the file system control unit 2 accesses the sector information table 12, acquires the physical block number having the largest number of invalid data sectors, and further accesses the block information table 10 to acquire the physical block number. The logical block number corresponding to the physical block number is acquired (step 151). Then, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4, and the block control sector 52a of the preliminary erase block 51 which is reserved in an unused state in advance for the rebuilding process of the file system. Is updated to the logical block number acquired in step 151, and the block state of the block control sector 52a is updated from "unused" to "data being transferred". Similarly, the block information table 10 is accessed, the logical block number of the preliminary erase block 51 is updated to the logical block number acquired in step 151, and the block state is updated from "unused" to "data transfer in progress" (step 152).

Next, the file system controller 2 accesses the sector information table 11 in order to efficiently copy the data from the erase block 51 having the physical block number acquired in step 151 to the spare erase block 51, and Of the physical sector numbers corresponding to the physical block number acquired in 151, the physical sector number whose sector status is "data valid" is searched and acquired (step 153). If there is a sector 52 whose sector status is “data valid” (step 154, Yes), the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4, and acquires the physical block number and the physical block number acquired in step 151. The sector state "data valid" of the block control sector 52a of the erase block 51 corresponding to the physical sector number acquired in step 153 is copied to the sector state of the block control sector 52a corresponding to the same physical sector number of the preliminary erase block 51. Similarly, the file system control unit 2 accesses the sector information table 11 and sets the physical block number acquired in step 151 and the sector state “data valid” of the physical sector number acquired in step 153 to the same physical block in the preliminary erase block 51. Copy to the sector status of the sector number. At the same time, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4, and from the sector 52 of the erase block 51 corresponding to the physical block number acquired in step 151 and the physical sector number acquired in step 153. The data is copied to the sector 52 of the same physical sector number of the preliminary erase block 51 (step 155).

The file system controller 2 confirms whether or not there is an unprocessed sector 52 in the erase block 51 of the physical block number acquired in step 151 (step 156). If there is an unprocessed sector 52, the process returns to step 153. If there is no unprocessed sector 52, the data copy is completed, and therefore the file system control unit 2 accesses the sector information table 12 and preliminarily erases the number of valid data sectors corresponding to the physical block number acquired in step 151. Copy to the number of data valid sectors in block 51, set the number of data invalid sectors to 0, and make the sum of the number of data valid sectors, the number of data invalid sectors, and the number of unused sectors to be the total number of sectors (127). The number of sectors is calculated and set (step 157).

Next, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4, and changes the block state of the block control sector 52a of the preliminary erase block 51 from "data transfer in progress" to "original block erase." To "medium" (step 158), erase the data of the erase block 51 having the physical block number acquired in step 151 (step 159), and change the block state of the block control sector 52a of the preliminary erase block 51 to "original block erase "To" "with data" is updated. Similarly, the file system control unit 2 accesses the block information table 10 to update the block state of the preliminary erase block 51 from "data transfer in progress" to "original block erase in progress" (step 158), and step 159 is performed. After that, the block state of the preliminary erase block 51 is updated from "source block is being erased" to "data is present". As a result, the block having the physical block number acquired in step 151, that is, the block from which the data has been erased is secured as the new preliminary erase block 51 (step 160).

Here, the block state of the block control sector 52a of the erase block 51 whose data has been erased in step 159 is automatically changed from "with data" or "block full" to "unused" by data erasing. Further, in the block information table 10, the sector information table 11, and the sector information table 12, the erase block 51 from which data has been erased in step 159 becomes a preliminary erase block, and the preliminary erase block is referred to when writing data to the flash memory section 5. Since it does not occur, it is not necessary to update the item corresponding to the physical block number of this preliminary erase block.

In the present embodiment, the process of reconstructing the file system 1 shows the case of erasing one erase block, but it is possible to execute the series of processes from step 151 a plurality of times. is there.

9A to 9D are flowcharts relating to the initialization of the file system 1 in this embodiment. This initialization is executed once when the file system 1 is started up, that is, when the power supply is restarted after the power supply is once cut off, and guarantees the reading of data from the flash memory unit 5.

First, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4, reads the block state of the block control sector 52a of each erase block 51, and based on the read block information, The block information of the block information table 10 is updated (step 171), it is confirmed whether or not the remaining erased blocks 51 exist (step 172), and the process of step 171 is repeated until the remaining erased blocks 51 are exhausted.

Next, the file system controller 2 accesses the block information table 10 updated in step 171, confirms whether the block status "data transfer in progress" exists (step 173), and checks the block status "data." If "in transfer" exists, the physical block number of this block state is acquired, and the block status corresponding to this physical block number is updated from "data transfer" to "unused" (step 174). At the same time, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4 and erases the data of the block having this physical block number.

Here, when the power supply is cut off in the middle of steps 151 to 157 of FIGS. 8A to 8B and the data transfer to the preliminary erasure block 51 is interrupted, the block state of the preliminary erasure block 51 is "data". "Transfer in progress". When the power supply is restarted after this, in the processing of steps 173 and 174, the data in the preliminary erase block 51 in the block state “data transfer in progress” in which the data transfer is interrupted is erased, and the block state of the preliminary erase block 51 It becomes "unused". This prevents incomplete data from being read from the preliminary erase block 51.

Next, the file system controller 2 accesses the block information table 10 updated in step 171, and confirms whether there is a block whose block status is "erasing original block" (step 175). If there is a block whose block state is “erasing original block”, the physical block number having the same logical block number as the logical block number of this block and the block state being “with data” is acquired. Then, the file system control unit 2 accesses the block information table 10 and updates the block state corresponding to this physical block number

to "unused". At the same time, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4 and erases the data of the block having this physical block number.

Also, the file system control unit 2 sets the block state information of the block control sector 52a of the erase block 51 having the physical block number whose block state acquired in step 175 is "erasing the original block" to "erasing the original block". To “with data”, and the block state corresponding to the same physical block number in the block information table 10 is updated to “with data” (step 176).

After that, the file system control unit 2 reads the sector state of the block control sector 52a of the erase block 51, and updates the sector information table 11 in the file system memory unit 3 based on this sector information (step 177).), it is confirmed whether or not the remaining sector information is present (step 178). If the remaining sector information is present, the process returns to step 177 and is repeated until there is no remaining sector 52. If there is any, the process returns to step 177 and is repeated until there are no more erase blocks 51 remaining.

Here, when the power supply is cut off in the middle of steps 158 to 160 of FIGS. 8A to 8B, the block state of the preliminary erase block 51 is held as "source block erasing in progress". Data transfer to the pre-erase block 51 has been completed. When the power supply is restarted after this, in the processing of steps 175 to 178, the data in the preliminary erase block 51 is held and the data in the erase block 51 of the data transfer source is erased. This prevents incomplete data from being read from the erase block 51 of the data transfer source.

Next, the file system controller 2 accesses the sector information table 11 updated in step 177, and confirms whether there is a sector 52 whose sector status is "data writing" (step 179). If there is a sector 52 whose sector status is "writing data", the physical block number and physical sector number corresponding to the sector status "writing data" acquired in step 179 are acquired. Then, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4, and sets the sector state of the block control sector 52a of the erase block 51 corresponding to these physical block number and physical sector number to "data writing". Update from "Medium" to "Data invalid". Similarly, the file system controller 2 accesses the sector information table 11 and updates the physical block number and the sector status corresponding to the physical sector number from "data writing" to "data invalid" (step 180). .

Next, the file system controller 2 accesses the sector information table 11 and confirms whether there is a sector 52 whose sector status is "data write completed" (step 181), and the sector status is " If there is a sector 52 for which “data writing is completed”, it is confirmed whether or not there is a sector 52 having the same logical sector number as that of the sector 52 and the sector state of “data valid” (step 182), if the sector 52 exists, the physical block number and the physical sector number of the sector 52 are acquired. Then, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4, and sets the sector state of the block control sector 52a of the erase block 51 corresponding to these physical block number and physical sector number to "data valid". Update from" to "Data invalid". Similarly, the file system controller 2 accesses the sector information table 11 and updates the sector state corresponding to these physical block numbers and physical sector numbers from "data valid" to "data invalid" (step 183).

Next, the file system controller 2 accesses the sector information table 11 and acquires the physical block number and the physical sector number of the sector 52 whose sector status acquired in step 181 is "data writing completed". Then, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4, and sets the sector state of the block control sector 52a of the erase block 51 corresponding to these physical blocks and physical sector numbers to "data writing". Update from “Complete” to “Data valid”. Similarly, the file system control unit 2 accesses the sector information table 11 and updates the sector state corresponding to these physical block numbers and physical sector numbers from "data write completion" to "data valid" (step 184). .

Here, when the power supply is cut off and the data writing is interrupted in the middle of steps 139 to 141 of FIGS. 7A to 7c, the sector state is held as “data writing”, and the data writing is performed. Is completed, the sector state is held as “data write completed”. When the power supply is resumed after this, in the processing of steps 179 to 184, the data of the sector 52 of the sector state "data writing" in which the data writing is interrupted is erased, and the sector state of this sector 52 becomes "data invalid". It will be. This prevents incomplete data from being read from the sector 52 in which data writing has been interrupted. Further, since the data of the sector 52 having the sector state “data writing completed” is held and the sector state is updated to “data valid”, the data can be correctly read from the sector 52.

Next, the file system control unit 2 accesses the sector information table 11 and, for each physical block number, the number of sector status "unused" and sector status "data valid" corresponding to the same physical block number. And the sector status "data invalid" are counted respectively, and based on these count values, the physical block number of the sector information table 12 in the file system memory unit 3, the number of unused sectors, the number of data valid sectors and the data invalid The number of sectors is updated (step 185), and it is confirmed whether there is a remaining physical block number (step 186). If there is a remaining physical block number, the process returns to step 185 and the remaining physical block number is changed. Repeat until it disappears.

After that, the file system control unit 2 determines whether the physical block number corresponding to the physical block number whose block status is "unused" is based on the block information table 10 updated in Step 171, Step 174, and Step 175. The erase block 51 is secured as the preliminary erase block 51 (step 187).

As described above, in this embodiment, when writing data or reconstructing a file system, the state information of each erase block 51 and each sector 52 is finely rewritten, so that the power supply may be cut off carelessly. When the power supply is restarted after this, the uncertain data in the erase block 51 or the sector 52 may be erased based on the state information of the erase block 51 or the sector 52, and only correct data may be saved. it can.

In the above embodiment, as the current block state, (1) 11111111b representing "unused" (b represents a binary number; the same applies hereinafter), (2) 11111110b representing "data being transferred", (5 types of block states are exemplified: 11111100b representing "erasing the original block", (4) 11111000b representing "data present", and (5) 1110000b representing "block full". It may be complicated or the number of bits may be increased, which makes it possible to more reliably determine the difference between the states.

Further, in order to manage the block state more finely, the number of blocks may be set to 6 or more, which allows the state to be more finely managed.

In the above embodiment, the current sector state is (I) 1111b indicating "unused", (II) 1110b indicating "data is being written", and (III) "data writing completed". 5 sector states 1100b, (IV) "data valid" 1000b, and (V) "data invalid" 0000b are illustrated. However, even if the combination of 1 and 0 is complicated or the number of bits is increased. Well, this makes it possible to more reliably determine the difference between the states.

Further, in order to manage the sector status more finely, the number of statuses of the sector 52 may be set to 6 or more, which makes it possible to manage the status more finely.

Although the flash memory having a uniform block size has been described in the above embodiment, any one erase block for a boot block (for example, a size of 8 Kbytes) can be used in a flash memory having a boot block configuration. (8 Kbytes) and any one erase block (64 Kbytes) for other blocks (for example, a size of 64 Kbytes) may be reserved in advance as a spare block for rebuilding the file system 1, This makes it possible to use a flash memory having a uniform block or boot block configuration, which is suitable for the purpose of the system, or a cheaper flash memory. Further, as a spare block for a boot block (for example, 8 Kbytes), it is also possible to allocate a block having a size larger than 8 Kbytes (for example, 64 Kbytes) to the spare block. Therefore, management becomes easy.

When the file system 1 of the present embodiment comprises a plurality of flash memories, one arbitrary erase block is secured in advance as a spare block for rebuilding the file system for each flash memory. Instead, reserve one spare block for multiple flash memories, or secure at least one spare block for each erase block with different block sizes included in multiple flash memories. In this case, the number of reserved blocks to be secured can be reduced.

In step 176 shown in FIG. 9B, the file system control unit 2 has the same logical block number as the logical block number of the erase block 51 in the block state "erasing the original block" acquired in step 175, and The physical block number whose block status is "with data" is acquired. Then, the file system control unit 2 accesses the block information table 10 and updates the block state corresponding to this physical block number to "unused". At the same time, the file system control unit 2 accesses the flash memory unit 5 through the flash memory control unit 4 and erases the data in the erase block 51 having this physical block number. Further, the file system control unit 2 changes the block state of the block control sector 52a of the erase block 51 having the physical block number of the block state “source block erasing” acquired in step 175 from “source block

erasing” to “data available”. , And the block status corresponding to the same physical block number in the block information table 10 is updated to “with data”.

Here, there are five states of the erased block 51, which are "unused", "data present", "block full", "data being transferred", and "source block being erased". In addition, as an example in the case where “original block erase completion” is set, the following process flow can be used.

FIG. 13 is a diagram showing a flow chart in which a portion corresponding to steps 174 to 177 in this embodiment is extracted from the flow chart relating to the initialization of the file system 1 shown in FIGS. 9A to 9D.

First, the file system controller 2 accesses the block information table 10 updated in step 171, confirms whether the block status "original block erase completed" exists (step 191), and checks the block status "original block". If "erase complete" exists, the process skips steps 192 and 193 and jumps to step 194.

Next, the file system controller 2 accesses the block information table 10 updated in step 171, confirms whether the block status "original block erased" exists (step 192), and checks the block status "original". If "erasing block" exists, the physical block number having the same logical block number as the logical block number of the erase block 51 and the block status being "with data" is acquired, and the file system controller 2 The flash memory unit 5 is accessed through the flash memory control unit 4 to erase the data in the erase block 51 having this physical block number.

After that, the file system control unit 2 changes the block state of the block control sector 52a of the erase block 51 of the physical block number whose block state acquired in step 192 is "erasing original block" to "erasing original block". "To erase the original block".

Further, the file system controller 2 accesses the block information table 10 and updates the block state corresponding to the physical block number of the erased block 51 from which data has been erased to "unused" (step 193).

Next, the block state of the block control sector 52a of the erase block 51 having the physical block number updated to the block state "original block erase complete" is updated from "original block erase complete" to "data present", and the block information is updated. The block state corresponding to this physical block number in Table 10 is updated from "source block being erased" to "data present" (step 194).

As described above, by adding the block state “original block erase completion”, it is possible to skip the block erasing process having a relatively long processing time, that is, the block state immediately before the interruption of the power supply is “original block erasing”. If it is immediately after the update from "during block erase" to "completion of original block erase", it is possible to shorten the initialization processing time when the power supply is restored.

In another embodiment, as shown in FIG. 12, a plurality of pieces of status information about the erase block 51 include "unused", "data present", "data being transferred", "source block being erased", It is also possible to set five of "in the original block format".

In this case, the means for updating the state information of the preliminary erase block 51 reserved for reconstruction from "unused" to "data being transferred" and the data in the erase block 51 of the data transfer source Means for transferring and storing the data to the preliminary erase block 51, and updating the block state information of the preliminary erase block 51 from "data being transferred" to "source block being deleted" after completion of transfer and storage of the data, The data in the erase block 51 of the data transfer source is erased, the erase block 51 of the data transfer source is secured as a new spare erase block 51, and the state information of the new spare erase block 51 is changed from "with data" to " Means for updating to "unused" and status information of the preliminary erase block 51 of the data transfer destination from "source block being erased" to "source block being formatted" to the new spare erase block 51 of the data transfer source. A means for writing (formatting) the number of erasable times of the new preliminary erase block 51 and the state information of the preliminary erase block 51 of the data transfer destination is updated from "source block being formatted" to "data present" after the formatting is completed. It is conceivable that the configuration is combined with the means for doing so.

As described above, five states of "unused", "data present", "data being transferred", "source block being erased", and "source block being formatted" were used as the state information of the erase block 51. In this case, it is possible to accurately grasp the state of the erase block and manage the flash memory in detail, without destroying the existing data stored in the flash memory and without causing inconsistency in the existing data. The file system can be automatically restored to the previous stage that was being executed immediately before the power supply was cut off.

In the initialization of the file system 1 at the time of power recovery in another embodiment, when the erase block 51 in the block state “source block erasing” is detected, the data in the erase block 51 of the data transfer source is erased. .. Alternatively, in order to further ensure the accuracy of the existing data, even when the erase block 51 of “source block erasing” as well as “source block erasing” is detected in the initialization, the erase block 51 of the data transfer source is not deleted. A method of erasing data can be considered.

Further, unlike the above-described embodiment in which the state "block full" is set as the state information of the erase block 51, in the method in which the block state "block full" is not set, the erase block is divided into sectors of uneven sizes. It is especially effective when used. This is because when writing data of unequal size to consecutive sectors in the same erase block, if the size of the data to be written next is unknown, the state information of the erase block becomes “block full”. This is because it is difficult to determine whether or not it is.

There is also a method of storing the number of erasable times of each erase block 51 in order to average the number of block erases of each erase block 51.

In another embodiment shown in FIGS. 8A and 8B, when the file system 1 is reconstructed, the file system control unit 2 excludes the preliminary erase block 51 as an erase block which is a data transfer source. If the difference between the maximum value and the minimum value of the erase counts exceeds a certain value and the erase block 51 has the maximum number of eraseable times, the erase block 51 having the maximum number of times is selected and the difference exceeds the certain value. If not, the sector information table 12 in the erase block 51 is accessed and the erase block 51 having the largest number of invalid data sectors is selected.

In this case, it is possible to suppress the variation in the number of erasable times of each erase block 51 within a fixed number and increase the number of times of rewriting as a flash memory.

Further, the number of erasable times of the erasing block 51 is incremented from 0 by 1 and incremented to the maximum value of the specification value of the number of erasable times (in this case, not the number of erasable times but the number of erasing). On the contrary, it is conceivable to decrease the maximum value of the erasable count by one from the maximum value and decrement it to 0. In the latter method, the erasure count becomes 0 when the erasable count reaches 0. Since it can be determined that the block 51 has reached the end of its life, the erasable count determination algorithm can be simplified.

The formatting means of the erasing block 51 in another embodiment can store the date and time of formatting, and in this case, the refresh operation of the data in the erasing block 51 can be performed after the elapse of a certain time. Become.

According to the file system of the present invention, in a file system using a nonvolatile semiconductor memory device having a plurality of erase blocks each including a plurality of sectors, an erase block is erased for each erase block. A storage unit that stores any one of the plurality of state information, and stores, for each sector, any one of the plurality of state information about the sector, and a storage unit in the storage unit when the nonvolatile semiconductor memory device is accessed. A guarantee means is provided for guaranteeing the data already stored in the nonvolatile semiconductor memory device based on the state information of each erase block and the state information of each sector.

According to the present invention, a plurality of state information about blocks and a plurality of state information about sectors are stored in the storage means. Therefore, even if the power supply is unintentionally cut off while the file system is operating, when the power supply is restored, a plurality of status information about blocks and a plurality of statuses about sectors at the time of initialization processing at startup. Based on the information, it is possible to accurately grasp the state of blocks and sectors and manage the flash memory in detail, without destroying the existing data stored in the flash memory and without causing inconsistency in the

existing data. It is possible to automatically restore the file system to the previous stage that was being executed immediately before the power supply was cut off, without causing it.

[0122] In one embodiment, the storage means is provided on a RAM that has no restriction on data updating, and the guarantee means is provided for each erase block on the RAM when the nonvolatile semiconductor storage device is accessed. The status information of each sector and the status information of each sector are referred to.

When the RAM is used in this way, there is no restriction in updating the data, and the processing speed is increased.

In one embodiment, at least five states of "unused", "data present", "block full", "data being transferred", and "original block being erased" are used as the plurality of status information about the erase block. A state is set, and the assurance means updates the state information of the spare block previously reserved for reconstruction from "unused" to "data being transferred", and the erase block in the data transfer source. Means for transferring and storing the data of the spare block to the spare block, and after the transfer and storage of the data are completed, the block status information of the spare block is updated from "data being transferred" to "erasing the original block". The data in the erase block of the data transfer source is erased, the erase block of the data transfer source is secured as a new spare block, and the status information of the new spare block is changed from "with data" or "block full" to "unused". And a means for updating the status information of the spare block of the data transfer destination from "erasing the original block" to "data present".

As described above, when the five states of "unused", "data present", "block full", "data being transferred", and "source block being erased" are used as the erase block state information, It is possible to accurately grasp the status and manage the flash memory in detail, and cut off the power supply without destroying the existing data stored in the flash memory and without causing inconsistency in the existing data. It is possible to automatically restore the file system to the previous stage that was being executed immediately before.

In one embodiment, at least one spare block previously reserved for reconstruction is provided for each erase block having a different block size included in one nonvolatile semiconductor memory device.

In this case, there is an advantage that it is possible to use a flash memory having a uniform block or boot block structure, which is suitable for the purpose of the system or is cheaper.

In one embodiment, at least one spare block previously reserved for reconstruction is provided for each erase block having a different block size included in a plurality of nonvolatile semiconductor memory devices.

In this case, there is an advantage that the number of spare blocks required by the file system can be reduced. Therefore, there is also an advantage that the file system can be constructed inexpensively.

In one embodiment, the spare block previously reserved for reconstruction has a size equal to or greater than the size of the erase block.

In this case, as the spare block for the boot block (for example, the size of 8 Kbytes), a block having a size larger than 8 Kbytes (for example, the size of 64 Kbytes) can be allocated to the spare block to make the size of the spare block constant. Therefore, management becomes easy.

The file system of the present invention is, in a file system using a nonvolatile semiconductor memory device having a plurality of sectors, a storage means for storing, for each sector, any one of a plurality of status information about the sector. A sector for guaranteeing data already stored in the nonvolatile semiconductor memory device based on the state information of each sector in the memory device when the nonvolatile semiconductor memory device is accessed. At least five states of "unused", "data writing", "data writing completed", "data valid", and "data invalid" are set as a plurality of state information about A means for updating the status information of the sector to which the data is written from "unused" to "data writing", and the status information of the sector from "data writing" to "data writing" after the data writing to the sector is completed. When data is newly written to the sector, the state information of the sector is updated from "data write completed" to "data valid", and the existing data in the sector is updated. When data is written, the status information of the other sector in which the existing data is written is updated from “data valid” to “data invalid”, and then the status information of the sector in which the updated data is written is updated. Update from "Data write completed" to "Data valid".

According to the present invention, the five status information about the sector is stored in the storage means. Therefore, even if the power supply is unintentionally cut off while the file system is operating, when the power supply is restored, the status of the sector can be determined based on multiple status information about the sector during the initialization process at startup. Just before shutting down the power supply, it is possible to accurately manage the flash memory and manage the flash memory in detail, without destroying the existing data stored in the flash memory and without causing inconsistency in the existing data. It is possible to automatically recover the file system to the previous stage that was being executed.

In one embodiment, the plural pieces of status information regarding the erase block include "unused", "data present", "block full", "data being transferred", "source block being erased", and "source block". At least six states of "erase completion" are set, and the guarantee means updates the state information of the preliminary erase block previously reserved for reconstruction from "unused" to "data transfer in progress". A means for transferring and storing the data in the erase block of the data transfer source to the preliminary erase block, and after completion of the transfer and storage of the data, the block state information of the preliminary erase block is changed from "data being transferred" to "source". "Erasing block", erasing the data in the erase block of the data transfer source, and updating the block state information of the preliminary erase block from "erasing the original block" to "erasing the original block", and data. A method of reserving the erase block of the transfer source as a new preliminary erase block and updating the status information of this new preliminary erase block from "with data" or "block full" to "unused" and the data transfer destination And means for changing the state information of the preliminary erase block from "original block erase completed" to "data present".

As described above, as the erase block status information, there are six states of "unused", "data present", "block full", "data being transferred", "source block being erased", and "source block erase completed". When the state is used, the state of the erase block can be grasped more accurately than in the case of using the five states, and the flash memory can be managed finely, and the existing data stored in the flash memory is destroyed. It is possible to automatically restore the file system to the previous stage that was being executed immediately before the power supply was cut off, without causing any inconvenience and without causing inconsistency in existing data.

In one embodiment, at least "unused", "data present", "data being transferred", "source block is being erased", and "source block is being formatted" as at least the plurality of state information about the erase block. Five states are set, and the assurance means updates the state information of the preliminary erase block secured in advance for reconstruction from "unused" to "data being transferred", and erases the data transfer source. A means for transferring and storing the data in the block to the preliminary erasing block, and the block state information of the preliminary erasing block from "data transfer" to "original block erasing" after completion of transfer and storage of the data The data in the erase block of the data transfer source is updated, the erase block of the data transfer source is secured as a new spare erase block, and the state information of the new spare erase block is changed from “with data” to “not available”. And the status information of the spare erase block of the data transfer destination is updated from "source block erasing" to "source block formatting", and a new spare erase block of the data transfer source is newly reserved. A means for writing (formatting) the erasable number of times of the erase block and the like, and a means for updating the status information of the preliminary erase method block of the data transfer destination from "source block being formatted" to "data present" after completion of formatting. Including.

In this way, when the five states of "unused", "data present", "data being transferred", "source block being erased", and "source block being formatted" are used as the erase block state information. , It is possible to accurately grasp the state of the erase block and manage the flash memory in detail, without destroying the existing data stored in the flash memory and without causing inconsistency in the existing data, It is possible to automatically restore the file system to the previous stage that was being executed immediately before the power supply was cut off.

In this case, unlike the above-described embodiment in which the state "block full" is set as the erase block state information, the method in which the block state "block full" is not set is that the erase block is divided into sectors of uneven sizes. It is especially effective when used. This is because when writing data of unequal size to consecutive sectors in the same erase block, if the size of the data to be written next is unknown, the state information of the erase block becomes “block full”. This is because it is difficult to determine whether or not it is.

In one embodiment, the erasable number of times of each erase block is stored in order to average the number of block erases of each erase block. By utilizing this erasable count when rebuilding the file system, it is possible to suppress the variation in the erasable count of each erase block within a fixed number and improve the total number of rewrites as a flash memory. .

Further, a method of increasing the erasable frequency of the erase block by 1 from 0 and incrementing it to the maximum value of the specification value of the erasable frequency (in this case, not the erasable frequency but the erasing frequency), On the contrary, a method of decreasing the erasable count by one from the maximum value and decrementing to 0 is conceivable. In the latter method, the erasure block is deleted when the erasable count reaches 0. Since it can be determined that the life of the erasure has come, the erasable count determination algorithm can be simplified.

The erasing block formatting means in one embodiment can also store the date and time of formatting, and in this case, the data in the erasing block can be refreshed after a certain period of time.

Brief description of the drawings

1 is a block diagram showing an embodiment of a file system of the present invention.

2 is a configuration diagram of the data stored in the flash memory unit of the file system of FIG.

3 is a data block diagram of each erase block stored in the file system memory unit in the file system of FIG.

4 is a data block diagram of each sector stored in the file system memory unit in the file system of FIG.

5 is a data configuration diagram in which the number of each sector is arranged according to the sector state for each erase block stored in the file system memory unit in the file system of FIG.

⟨Figure 6 ⟩It is the flowchart regarding the data read processing in the file system in Figure 1.

7C is a flowchart following FIG. 7B regarding the process of writing data in the file system of FIG. 1.

9C is a flowchart subsequent to FIG. 9B regarding initialization of the file system of FIG. 1.

9D is a flowchart following FIG. 9C regarding initialization of the file system of FIG. 1.

⟨Figure 10 ⟩It is the flowchart regarding the renewal processing of the data which is housed in the flash memory in the former technology.

⟨Figure 11 ⟩It is the flowchart regarding the rebuilding processing of the file system in the former technology.

12 is a diagram showing another embodiment showing the block state of the block state information in the configuration diagram of the data stored in the flash memory unit in the file system of FIG. 1.

13 is a diagram showing a flowchart of a portion related to the first embodiment in which "original block erase completion" is set as the state information in FIGS. 9A to 9D regarding initialization of the file system of FIG.

Explanation of symbols

1 File System 2 File System Control Unit 3 File System Memory Unit 4 Flash Memory Control Unit 5 Flash Memory Unit 6 Application, OS (Operating System) 51 Erase Block, Pre-Erase Block 52 Sector 52a Block Control Sector

(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2001－51889
（P2001－51889A）
(43)公開日　平成13年２月23日(2001.2.23)

(51)Int.Cl.[7]	識別記号	ＦＩ		テーマコード（参考）
Ｇ０６Ｆ　12/00	５４２	Ｃ０６Ｆ　12/00	５４２Ｍ	５Ｂ０１８
12/16	３４０	12/16	３４０Ｐ	５Ｂ０８２

審査請求　未請求　請求項の数19　ＯＬ　（全 23 頁）

(21)出願番号　特願平11－254973

(22)出願日　平成11年９月８日(1999.9.8)

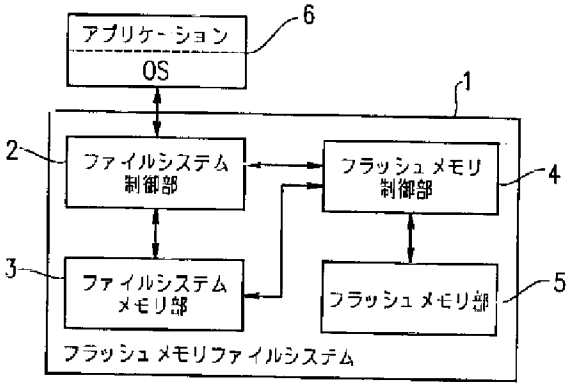
(31)優先権主張番号　特願平11－153090
(32)優先日　平成11年５月31日(1999.5.31)
(33)優先権主張国　日本（ＪＰ）

(71)出願人　000005049
シャープ株式会社
大阪府大阪市阿倍野区長池町22番22号
(72)発明者　戸森　宣昭
大阪府大阪市阿倍野区長池町22番22号　シャープ株式会社内
(74)代理人　100078282
弁理士　山本　秀策

Ｆターム(参考)　5B018 GA04 LA06 MA22 NA06 QA05
QA15
5B082 JA07

(54)【発明の名称】　不揮発性半導体記憶装置を用いたファイルシステム

(57)【要約】
【課題】　動作中に意図しない電力供給の遮断が発生しても、不揮発性半導体記憶装置内に格納されているデータを正常に読み出すことを可能にする。
【解決手段】　各消去ブロック毎に、消去ブロックについての複数の状態情報のいずれかを格納すると共に、該各セクタ毎に、セクタについての複数の状態情報のいずれかをファイルシステムメモリ部３及びフラッシュメモリ部５に格納している。ファイルシステムメモリ部３内の各消去ブロックの状態情報及び各セクタの状態情報は、ファイルシステム１の起動時に、フラッシュメモリ部５内の該データを読み出すことによって作成され、ファイルシステム制御部２は、ファイルシステムメモリ部３内の各消去ブロックの状態情報及び各セクタの状態情報に基いて、フラッシュメモリ部５からデータを読み出すと共に、動作中に意図しない電力供給の遮断時においてもフラッシュメモリ部５内のデータを保証する。



SAMSUNG EX1002, Page 400 of 1031

【特許請求の範囲】
【請求項１】　複数のセクタをそれぞれ含む複数の消去ブロックを有する不揮発性半導体記憶装置を用いたファイルシステムにおいて、
該各消去ブロック毎に、消去ブロックについての複数の状態情報のいずれかを格納すると共に、該各セクタ毎に、セクタについての複数の状態情報のいずれかを格納する記憶手段と、
該不揮発性半導体記憶装置のアクセス時に、該記憶手段内の該各消去ブロックの状態情報及び該各セクタの状態情報に基いて、該不揮発性半導体記憶装置に既に格納されているデータを保証する保証手段とを具備する不揮発性半導体記憶装置を用いたファイルシステム。
【請求項２】　前記記憶手段は、データの更新に制約の無いＲＡＭ上に設けられ、
前記保証手段は、前記不揮発性半導体記憶装置へのアクセス時に、該ＲＡＭ上の前記各消去ブロックの状態情報及び前記各セクタの状態情報を参照する請求項１に記載の不揮発性半導体記憶装置を用いたファイルシステム。
【請求項３】　前記消去ブロックについての複数の状態情報として、「未使用」、「データ有」、「ブロックフル」、「データ転送中」、「元ブロック消去中」という少なくとも５つの状態を設定しており、
前記保証手段は、
再構築用に予め確保されている予備ブロックの状態情報を「未使用」から「データ転送中」へ更新する手段と、データ転送元の消去ブロック内のデータを該予備ブロックへ転送して記憶させる手段と、
該データの転送並びに記憶の完了後、該予備ブロックのブロック状態情報を「データ転送中」から「元ブロック消去中」へ更新し、該データ転送元の消去ブロックのデータを消去し、このデータ転送元の消去ブロックを新たな予備ブロックとして確保し、この新たな予備ブロックの状態情報を「データ有」あるいは「ブロックフル」から「未使用」へ更新する手段と、
データ転送先の該予備ブロックの状態情報を「元ブロック消去中」から「データ有」へ更新する手段とを含む請求項１に記載の不揮発性半導体記憶装置を用いたファイルシステム。
【請求項４】　ファイルシステムの再構築用に予め確保されている予備ブロックは、１個の不揮発性半導体記憶装置に含まれるブロックサイズが異なる各消去ブロック毎に、少なくとも１個以上有する請求項３に記載の不揮発性半導体記憶装置を用いたファイルシステム。
【請求項５】　ファイルシステムの再構築用に予め確保されている予備ブロックは、複数個の不揮発性半導体記憶装置に含まれるブロックサイズが異なる各消去ブロック毎に、少なくとも１個以上有する請求項３に記載の不揮発性半導体記憶装置を用いたファイルシステム。
【請求項６】　ファイルシステムの再構築用に予め確保されている予備ブロックは、消去ブロックのサイズ以上のサイズを有する請求項４又は５に記載の不揮発性半導体記憶装置を用いたファイルシステム。
【請求項７】　複数のセクタを有する不揮発性半導体記憶装置を用いたファイルシステムにおいて、
該各セクタ毎に、セクタについての複数の状態情報のいずれかを格納する記憶手段と、
該不揮発性半導体記憶装置のアクセス時に、該記憶手段内の該各セクタの状態情報に基いて、該不揮発性半導体記憶装置に既に格納されているデータを保証する保証手段とを具備し、
前記セクタについての複数の状態情報として、「未使用」、「データ書き込み中」、「データ書き込み完了」、「データ有効」、「データ無効」という少なくとも５つの状態を設定しており、
前記保証手段は、
データが書き込まれるセクタの状態情報を「未使用」から「データ書き込み中」へ更新する手段と、
該セクタへのデータの書き込みが完了した後、該セクタの状態情報を「データ書き込み中」から「データ書き込み完了」へ更新する手段と、
該セクタにデータを新規に書き込んだ場合は、該セクタの状態情報を「データ書き込み完了」から「データ有効」へ更新し、該セクタに既存のデータの更新データを書き込んだ場合は、該既存のデータが書き込まれている他のセクタの状態情報を「データ有効」から「データ無効」に更新後、該更新データが書き込まれている該セクタの状態情報を「データ書き込み完了」から「データ有効」へ更新する不揮発性半導体記憶装置を用いたファイルシステム。
【請求項８】　電力供給が回復した時のファイルシステムの初期化において、前記消去ブロックの状態情報として「元ブロック消去中」の消去ブロックが存在するか検索し、該消去ブロックが存在する場合は、該消去ブロックの論理ブロック番号と同一の論理ブロック番号を有し、消去ブロックの状態情報が「データ有」あるいは「ブロックフル」である消去ブロックを、前記記憶手段内の各消去ブロックの状態情報から取得する手段と、取得した消去ブロックのデータを消去し、消去完了後、この消去ブロックの状態情報を「データ有」あるいは「ブロックフル」から「未使用」へ更新する手段と、消去ブロックの状態情報が「元ブロック消去中」の消去ブロックの状態情報を「元ブロック消去中」から「データ有」へ更新する手段とを含む請求項３に記載の不揮発性半導体記憶装置を用いたファイルシステム。
【請求項９】　前記消去ブロックについての複数の状態情報として、「未使用」、「データ有」、「データ転送中」、「元ブロック消去中」、「元ブロックフォーマット中」という少なくとも５つの状態を設定しており、前記保証手段は、ファイルシステムの再構築用に予め確保

されている予備消去ブロックの状態情報を「未使用」から「データ転送中」へ更新する手段と、データ転送元の消去ブロック内のデータを該予備消去ブロックへ転送して記憶させる手段と、該データの転送並びに記憶の完了後、該予備消去ブロックの状態情報を「データ転送中」から「元ブロック消去中」へ更新し、データ転送元の消去ブロックのデータを消去し、消去完了後、このデータ転送元の消去ブロックを新たな予備消去ブロックとして確保し、この新たな予備消去ブロックの状態情報を「データ有」から「未使用」へ更新する手段と、データ転送先の該予備消去ブロックの状態情報を「元ブロック消去中」から「元ブロックフォーマット中」へ更新し、データ転送元の新たな予備消去ブロックをフォーマットする手段と、フォーマット完了後、データ転送先の該予備消去ブロックの状態情報を「元ブロックフォーマット中」から「データ有」へ更新する手段とを含む請求項１に記載の不揮発性半導体記憶装置を用いたファイルシステム。

【請求項１０】　前記消去ブロックは、ブロック消去可能回数を各消去ブロック内に記憶しており、前記フォーマット時において、新たな予備消去ブロックに該予備消去ブロックの消去可能回数から１減じた回数を該予備消去ブロック内に記憶する手段と、ファイルシステムの再構築時には、該予備消去ブロックを除いた全ての消去ブロックの中で、ブロック消去回数が最大の消去ブロックをデータ転送元として選択する手段とを含む請求項９に記載の不揮発性半導体記憶装置を用いたファイルシステム。

【請求項１１】　前記ブロック消去可能回数が０以下の値になった時点で、該消去ブロックの使用を禁止する請求項１０に記載の不揮発性半導体記憶装置を用いたファイルシステム。

【請求項１２】　前記消去ブロックは、ブロック消去可能回数を各消去ブロック内に記憶しており、前記フォーマット時において、新たな予備消去ブロックに該予備消去ブロックの消去回数に１増加した回数を該予備消去ブロック内に記憶する手段と、ファイルシステムの再構築時には、該予備消去ブロックを除いた全ての消去ブロックの中で、ブロック消去回数が最大の消去ブロックをデータ転送元として選択する手段とを含む請求項９に記載の不揮発性半導体記憶装置を用いたファイルシステム。

【請求項１３】　前記ブロック消去可能回数が消去可能回数の仕様値（最大値）以上の値になった時点で、該消去ブロックの使用を禁止する請求項１２に記載の不揮発性半導体記憶装置を用いたファイルシステム。

【請求項１４】　ファイルシステムの再構築用に予め確保されている予備消去ブロックは、１個の不揮発性半導体記憶装置に含まれるブロックサイズが異なる各消去ブロック毎に、少なくとも１個以上有する請求項９に記載の不揮発性半導体記憶装置を用いたファイルシステム。

【請求項１５】　ファイルシステムの再構築用に予め確保されている予備消去ブロックは、複数個の不揮発性半導体記憶装置に含まれるブロックサイズが異なる各消去ブロック毎に、少なくとも１個以上有する請求項９に記載の不揮発性半導体記憶装置を用いたファイルシステム。

【請求項１６】　ファイルシステムの再構築用に予め確保されている予備消去ブロックは、消去ブロックサイズ以上のサイズを有する請求項１４または１５に記載の不揮発性半導体記憶装置を用いたファイルシステム。

【請求項１７】　複数のセクタを有する不揮発性半導体記憶装置を用いたファイルシステムにおいて、該各セクタ毎に、セクタについての複数の状態情報のいずれかを格納する記憶手段と、該不揮発性半導体記憶装置へのアクセス時に、該記憶手段内の該各セクタの状態情報に基いて、該不揮発性半導体記憶装置に既に格納されているデータを保証する保証手段とを具備し、前記セクタについての複数の状態情報として、「未使用」、「データ書き込み中」、「データ書き込み完了」、「データ有効」、「データ無効」という少なくとも５つの状態を設定しており、前記保証手段は、データが書き込まれるセクタの状態情報を「未使用」から「データ書き込み中」へ更新する手段と、該セクタへのデータの書き込みが完了した後、該セクタの状態情報を「データ書き込み中」から「データ書き込み完了」へ更新する手段と、該セクタにデータを新規に書き込んだ場合は、該セクタの状態情報を「データ書き込み完了」から「データ有効」へ更新し、該セクタに既存のデータの更新データを書き込んだ場合は、該既存のデータが書き込まれている他のセクタの状態情報を「データ有効」から「データ無効」に更新後、該更新データが書き込まれている該セクタの状態情報を「データ書き込み完了」から「データ有効」へ更新する手段とを含む請求項９に記載の不揮発性半導体記憶装置を用いたファイルシステム。

【請求項１８】　電力供給が回復した時のファイルシステムの初期化において、前記消去ブロックの状態情報として「元ブロック消去中」の消去ブロックが存在するか検索し、該消去ブロックが存在する場合は、該消去ブロックへのデータ転送元ブロックを、前記記憶手段内の各消去ブロックの状態情報から取得する手段と、取得したデータ転送元ブロックのデータを消去し、消去完了後、この消去ブロックの状態情報を「データ有」から「未使用」へ更新する手段と、消去ブロックの状態情報が「元ブロック消去中」の消去ブロックの状態情報を「元ブロック消去中」から「元ブロックフォーマット中」へ更新し、データを消去した消去ブロックをフォーマットして新たな予備消去ブロックとして確保する手段と、フォーマット完了後、消去ブロックの状態情報が「元ブロックフォーマット中」の消去ブロックの状態情報を「元ブロックフォーマット中」から「データ有」へ更新する手段

SAMSUNG EX1002, Page 402 of 1031

とを含む請求項９に記載の不揮発性半導体記憶装置を用いたファイルシステム。

【請求項１９】　電力供給が回復した時のファイルシステムの初期化において、前記消去ブロックの状態情報として「元ブロックフォーマット中」の消去ブロックが存在するか検索し、該消去ブロックが存在する場合は、該消去ブロックへのデータ転送元ブロックを、前記記憶手段内の各消去ブロックの状態情報から取得する手段と、取得したデータ転送元ブロックをフォーマットして新たな予備消去ブロックとして確保する手段と、フォーマット完了後、消去ブロックの状態情報が「元ブロックフォーマット中」の消去ブロックの状態情報を「元ブロックフォーマット中」から「データ有」へ更新する手段とを含む請求項９に記載の不揮発性半導体記憶装置を用いたファイルシステム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、フラッシュメモリ等の不揮発性半導体記憶装置を用いたファイルシステムに関するものである。

【０００２】

【従来の技術】最近、データを電気的に書き込み及び消去することが可能な不揮発性半導体記憶装置（例えばフラッシュメモリ）を、データを磁気的に書き込み及び消去することが可能なハードディスクドライブの代替として用いるための技術が注目されている。

【０００３】フラッシュメモリは、ハードディスクドライブと比較して、データの読み出し速度が約１万倍程度かつデータの書き込み速度が約１千倍程度速く、また小型軽量化を図り易く耐衝撃性に優れ、可搬性が高い等の特長を備え、更に低容量のデータを高速に格納しかつ読み出すという用途には好適である。

【０００４】しかしながら、フラッシュメモリにおいては、既にデータが書き込まれている記憶領域に新しくデータを書き込むには、書き込まれているデータを一旦消去するという動作が必要となる。しかも、データを消去する単位が数キロバイトと大きく、数十ミリ秒から数秒と比較的長いデータ消去時間が必要であり、データの書き込み及び消去の保証回数が有限であるという欠点がある。

【０００５】このため、フラッシュメモリをハードディスクドライブの代替として用いる場合は、上記欠点を踏まえ、データの取り扱い単位をハードディスクドライブ並みの数百バイト程度に小さくし、なるべくデータ消去を行なわず、効率良くデータを管理するための仕組み、つまりフラッシュメモリに対応したファイルシステムを提供する必要がある。この様なファイルシステムは、例えば特開平６－９５９５５号公報に開示されている。

【０００６】また、米国特許５，５４４，１１９号には、電力供給遮断後の電力復旧時の処理方法が示され、米国特許５，５４４，３５６号には、ファイルシステムの再構築方法が示されている。

【０００７】

【発明が解決しようとする課題】しかしながら、上記公報に記載の技術においては、電源として電池等を用いた電力供給が不安定な環境下で、フラッシュメモリや各種の制御を行うファイルシステムが使用される状況を想定しておらず、フラッシュメモリあるいはファイルシステムの動作中に、電池の消耗等により電力供給が一瞬でも遮断されたとき、つまり意図しない電力供給の遮断が発生したときには、フラッシュメモリに格納されているデータを正常に読み出せなくなるという可能性があった。

【０００８】具体的には、第１に、データが格納された記憶領域（以下セクタと呼称する）の現在の状態をセクタ状態情報によって表しているが、このセクタ状態情報が「未使用」、「使用中」、「無効」という３種類の状態のいずれかを表しているに過ぎないため、セクタのデータを更新するに際し、電力供給の遮断が発生すると、データが正常に更新されなくても、セクタ状態情報に基いてデータを正常とみなしてしまうことがあった。このセクタのデータの更新処理を図１０のフローチャートに従って次に述べる。

【０００９】図１０のフローチャートによれば、まず更新される既存のデータを格納している１つのセクタのセクタ状態情報を「使用中」から「無効」へ変更し（ステップ１０１）、次にもう１つのセクタのセクタ状態情報を「未使用」から「使用中」へ変更し（ステップ１０２）、この後にセクタ状態情報が「使用中」となったセクタに対して更新データを書き込む（ステップ１０３）。ステップ１０１～１０３を書き込むデータがなくなるまで繰り返す。

【００１０】ここで、ステップ１０２の直後あるいはステップ１０３の実行中において、電力供給の遮断が発生し、フラッシュメモリに対するデータの書き込みが中断すると、更新データのセクタへの書き込みが正常に終了していないにもかかわらず、このセクタのセクタ状態情報が「使用中」となる。このため、電力供給が回復したときに、ファイルシステムは、このセクタに格納されているデータを有効であると誤認してしまう。

【００１１】第２に、１つのブロック内の全てのデータを消去して、ファイルシステムを再構築するに際し、電源の遮断が発生すると、２つのブロックにおいて同じデータを格納した状態、あるいは１つのブロック内の全てのデータを完全に消去していない状態が発生することがあった。このファイルシステムの再構築の処理を図１１のフローチャートに従って次に述べる。

【００１２】図１１のフローチャートによれば、まず全てのセクタが消去され、全てのセクタのセクタ状態情報が「未使用」である予備ブロックを予め確保しておき、全てのデータが消去されるブロックにおける保存すべき

データ、つまりセクタ状態情報が「無効」であるセクタのデータ及びセクタ状態情報を除く他の全てのデータを予備ブロックにコピーし（ステップ１１１）、更に全てのデータが消去されるブロックにおけるセクタ状態情報が「無効」であるセクタ以外の位置情報に含まれるブロック番号を予備ブロックのブロック番号に更新する（ステップ１１２）。最後に、データのコピー元であるブロックを消去し、次回のファイルシステム再構築時の予備ブロックとして確保する（ステップ１１３）。

【００１３】ここで、ステップ１１３の実行前あるいはステップ１１３の実行中において、電力供給の遮断が発生すると、電力供給が回復したときに、２つのブロックにおいて同じデータを格納した状態、あるいは１つのブロック内の全てのデータを完全に消去できていない状態が発生し、新たな予備ブロックを確保できない。

【００１４】そこで、本発明は、上記従来の問題に鑑みなされたものであり、電力供給の遮断が発生しても、フラッシュメモリに格納されているデータを正常に読み出すことが可能な不揮発性半導体記憶装置を用いたファイルシステムを提供することを目的とする。

【００１５】

【課題を解決するための手段】上記課題を解決するために、本発明のファイルシステムは、複数のセクタをそれぞれ含む複数の消去ブロックを有する不揮発性半導体記憶装置を用いたファイルシステムにおいて、該各消去ブロック毎に、消去ブロックについての複数の状態情報のいずれかを格納すると共に、該各セクタ毎に、セクタについての複数の状態情報のいずれかを格納する記憶手段と、該不揮発性半導体記憶装置のアクセス時に、該記憶手段内の該各消去ブロックの状態情報及び該各セクタの状態情報に基いて、該不揮発性半導体記憶装置に既に格納されているデータを保証する保証手段とを具備する。

【００１６】本発明によれば、ブロックについての複数の状態情報及びセクタについての複数の状態情報を記憶手段に格納している。このため、ファイルシステムが稼働中において、意図しない電力供給の遮断が生じても、電力供給が回復したときには、起動時の初期化処理に際し、ブロックについての複数の状態情報及びセクタについての複数の状態情報に基いて、ブロック及びセクタの状態を正確に把握してきめ細かにフラッシュメモリを管理することができ、フラッシュメモリに格納されている既存のデータを破壊することなく、また既存のデータに矛盾を生じさせることなく、電力供給の遮断直前に実行中であった前段階へファイルシステムを自動的に回復させることが可能となる。

【００１７】一実施形態では、前記記憶手段は、データの更新に制約の無いＲＡＭ上に設けられ、前記保証手段は、前記不揮発性半導体記憶装置へのアクセス時に、該ＲＡＭ上の前記各消去ブロックの状態情報及び前記各セクタの状態情報を参照する。

【００１８】この様にＲＡＭを利用すると、データの更新に制約がなくなり処理速度が早くなる。

【００１９】一実施形態では、前記消去ブロックについての複数の状態情報として、「未使用」、「データ有」、「ブロックフル」、「データ転送中」、「元ブロック消去中」という少なくとも５つの状態を設定しており、前記保証手段は、再構築用に予め確保されている予備ブロックの状態情報を「未使用」から「データ転送中」へ更新する手段と、データ転送元の消去ブロック内のデータを該予備ブロックへ転送して記憶させる手段と、該データの転送並びに記憶の完了後、該予備ブロックのブロック状態情報を「データ転送中」から「元ブロック消去中」へ更新し、該データ転送元の消去ブロックのデータを消去し、このデータ転送元の消去ブロックを新たな予備ブロックとして確保し、この新たな予備ブロックの状態情報を「データ有」あるいは「ブロックフル」から「未使用」へ更新する手段と、データ転送先の該予備ブロックの状態情報を「元ブロック消去中」から「データ有」へ更新する手段とを含む。

【００２０】この様に消去ブロックの状態情報として「未使用」、「データ有」、「ブロックフル」、「データ転送中」、「元ブロック消去中」という５つの状態を利用した場合、ブロックの状態を正確に把握してきめ細かにフラッシュメモリを管理することができ、フラッシュメモリに格納されている既存のデータを破壊することなく、また既存のデータに矛盾を生じさせることなく、電力供給の遮断直前に実行中であった前段階へファイルシステムを自動的に回復させることが可能となる。

【００２１】一実施形態では、再構築用に予め確保されている予備ブロックは、１個の不揮発性半導体記憶装置に含まれるブロックサイズが異なる各消去ブロック毎に、少なくとも１個以上有する。

【００２２】この場合、均等ブロックあるいはブートブロック構成のフラッシュメモリのうちからシステムの目的に合わせた、あるいはより安価なフラッシュメモリを採用することが可能となる利点がある。

【００２３】一実施形態では、再構築用に予め確保されている予備ブロックは、複数個の不揮発性半導体記憶装置に含まれるブロックサイズが異なる各消去ブロック毎に、少なくとも１個以上有する。

【００２４】この場合、ファイルシステムが必要とする予備ブロックの数を少なくできる利点がある。このため、ファイルシステムが安価に構成できる利点もある。

【００２５】一実施形態では、再構築用に予め確保されている予備ブロックは、消去ブロックのサイズ以上のサイズを有する。

【００２６】この場合は、ブートブロック（例えば８Ｋバイトのサイズ）用の予備ブロックとして、８Ｋバイトより大きいサイズのブロック（例えば６４Ｋバイトのサイズ）を予備ブロックに割り当て、予備ブロックのサイ

ズを一定にできるため、管理が簡単になる。

【００２７】本発明のファイルシステムは、複数のセクタを有する不揮発性半導体記憶装置を用いたファイルシステムにおいて、該各セクタ毎に、セクタについての複数の状態情報のいずれかを格納する記憶手段と、該不揮発性半導体記憶装置のアクセス時に、該記憶手段内の該各セクタの状態情報に基いて、該不揮発性半導体記憶装置に既に格納されているデータを保証する保証手段とを具備し、前記セクタについての複数の状態情報として、「未使用」、「データ書き込み中」、「データ書き込み完了」、「データ有効」、「データ無効」という少なくとも５つの状態を設定しており、前記保証手段は、データが書き込まれるセクタの状態情報を「未使用」から「データ書き込み中」へ更新する手段と、該セクタへのデータの書き込みが完了した後、該セクタの状態情報を「データ書き込み中」から「データ書き込み完了」へ更新する手段と、該セクタにデータを新規に書き込んだ場合は、該セクタの状態情報を「データ書き込み完了」から「データ有効」へ更新し、該セクタに既存のデータの更新データを書き込んだ場合は、該既存のデータが書き込まれている他のセクタの状態情報を「データ有効」から「データ無効」に更新後、該更新データが書き込まれている該セクタの状態情報を「データ書き込み完了」から「データ有効」へ更新する。

【００２８】本発明によれば、セクタについての５つの状態情報を記憶手段に格納している。このため、ファイルシステムが稼働中において、意図しない電力供給の遮断が生じても、電力供給が回復したときには、起動時の初期化処理に際し、セクタについての複数の状態情報に基いて、セクタの状態を正確に把握してきめ細かにフラッシュメモリを管理することができ、フラッシュメモリに格納されている既存のデータを破壊することなく、また既存のデータに矛盾を生じさせることなく、電力供給の遮断直前に実行中であった前段階へファイルシステムを自動的に回復させることが可能となる。

【００２９】他の実施形態では、前記消去ブロックについての複数の状態情報として、「未使用」、「データ有」、「データ転送中」、「元ブロック消去中」、「元ブロックフォーマット中」という少なくとも５つの状態を設定しても良い。

【００３０】この場合、再構築用に予め確保されている予備消去ブロックの状態情報を「未使用」から「データ転送中」へ更新する手段と、データ転送元の消去ブロック内のデータを該予備消去ブロックへ転送して記憶させる手段と、該データの転送並びに記憶の完了後、該予備消去ブロックのブロック状態情報を「データ転送中」から「元ブロック消去中」へ更新し、該データ転送元の消去ブロックのデータを消去し、このデータ転送元の消去ブロックを新たな予備消去ブロックとして確保し、この新たな予備消去ブロックの状態情報を「データ有」から「未使用」へ更新する手段と、データ転送先の該予備消去ブロックの状態情報を「元ブロック消去中」から「元ブロックフォーマット中」へ更新し、データ転送元の新たな予備消去ブロックへ新たな予備消去ブロックの消去可能回数などを書き込む（フォーマットする）手段と、フォーマット完了後、データ転送先の該予備消去ブロックの状態情報を「元ブロックフォーマット中」から「データ有」へ更新する手段とを含む構成が考えられる。

【００３１】この様に、消去ブロックの状態情報として「未使用」、「データ有」、「データ転送中」、「元ブロック消去中」、 元ブロックフォーマット中」という５つの状態を利用した場合、消去ブロックの状態を正確に把握してきめ細かにフラッシュメモリを管理することができ、フラッシュメモリに格納されている既存のデータを破壊することなく、また既存のデータに矛盾を生じさせることなく、電力供給の遮断直前に実行中であった前段階へファイルシステムを自動的に回復させることが可能となる。また、消去ブロックの状態情報として「ブロックフル」という状態を設定しない方法は、消去ブロックが不均等なサイズのセクタに分割されて使用される場合に特に有効である。なぜならば、不均等なサイズのデータを同一の消去ブロック内の連続したセクタに書き込む場合において、次に書き込むべきデータのサイズが不明である場合に、消去ブロックの状態情報が「ブロックフル」となるかどうかの判断が難しいためである。

【００３２】電力供給が回復した時のファイルソステムの初期化において、前記消去ブロックの状態情報として「元ブロック消去中」の消去ブロックが存在するか検索し、該消去ブロックが存在する場合は、該消去ブロックの論理ブロック番号と同一の論理ブロック番号を有し、消去ブロックの状態情報が「データ有」あるいは「ブロックフル」である消去ブロックを、前記記憶手段内の各消去ブロックの状態情報から取得する手段と、取得した消去ブロックのデータを消去し、消去完了後、この消去ブロックの状態情報を「データ有」あるいは「ブロックフル」から「未使用」へ更新する手段と、消去ブロックの状態情報が「元ブロック消去中」の消去ブロックの状態情報を「元ブロック消去中」から「データ有」へ更新する手段とを含んでもよい。

【００３３】前記消去ブロックについての複数の状態情報として、「未使用」、「データ有」、「データ転送中」、「元ブロック消去中」、「元ブロックフォーマット中」という少なくとも５つの状態を設定しており、前記保証手段は、ファイルシステムの再構築用に予め確保されている予備消去ブロックの状態情報を「未使用」から「データ転送中」へ更新する手段と、データ転送元の消去ブロック内のデータを該予備消去ブロックへ転送して記憶させる手段と、該データの転送並びに記憶の完了後、該予備消去ブロックの状態情報を「データ転送中」から「元ブロック消去中」へ更新し、データ転送元の消

去ブロックのデータを消去し、消去完了後、このデータ転送元の消去ブロックを新たな予備消去ブロックとして確保し、この新たな予備消去ブロックの状態情報を「データ有」から「未使用」へ更新する手段と、データ転送先の該予備消去ブロックの状態情報を「元ブロック消去中」から「元ブロックフォーマット中」へ更新し、データ転送元の新たな予備消去ブロックをフォーマットする手段と、フォーマット完了後、データ転送先の該予備消去ブロックの状態情報を「元ブロックフォーマット中」から「データ有」へ更新する手段とを含んでもよい。

【００３４】前記消去ブロックは、ブロック消去可能回数を各消去ブロック内に記憶しており、前記フォーマット時において、新たな予備消去ブロックに該予備消去ブロックの消去可能回数から１減じた回数を該予備消去ブロック内に記憶する手段と、ファイルシステムの再構築時には、該予備消去ブロックを除いた全ての消去ブロックの中で、ブロック消去回数が最大の消去ブロックをデータ転送元として選択する手段とを含んでもよい。

【００３５】前記ブロック消去可能回数が０以下の値になった時点で、該消去ブロックの使用を禁止してもよい。

【００３６】前記消去ブロックは、ブロック消去可能回数を各消去ブロック内に記憶しており、前記フォーマット時において、新たな予備消去ブロックに該予備消去ブロックの消去回数に１増加した回数を該予備消去ブロック内に記憶する手段と、ファイルシステムの再構築時には、該予備消去ブロックを除いた全ての消去ブロックの中で、ブロック消去回数が最大の消去ブロックをデータ転送元として選択する手段とを含んでもよい。

【００３７】前記ブロック消去可能回数が消去可能回数の仕様値（最大値）以上の値になった時点で、該消去ブロックの使用を禁止してもよい。

【００３８】ファイルシステムの再構築用に予め確保されている予備消去ブロックは、１個の不揮発性半導体記憶装置に含まれるブロックサイズが異なる各消去ブロック毎に、少なくとも１個以上有してもよい。

【００３９】ファイルシステムの再構築用に予め確保されている予備消去ブロックは、複数個の不揮発性半導体記憶装置に含まれるブロックサイズが異なる各消去ブロック毎に、少なくとも１個以上有してもよい。

【００４０】ファイルシステムの再構築用に予め確保されている予備消去ブロックは、消去ブロックサイズ以上のサイズを有してもよい。

【００４１】複数のセクタを有する不揮発性半導体記憶装置を用いたファイルシステムにおいて、該各セクタ毎に、セクタについての複数の状態情報のいずれかを格納する記憶手段と、該不揮発性半導体記憶装置へのアクセス時に、該記憶手段内の該各セクタの状態情報に基いて、該不揮発性半導体記憶装置に既に格納されているデータを保証する保証手段とを具備し、前記セクタについての複数の状態情報として、「未使用」、「データ書き込み中」、「データ書き込み完了」、「データ有効」、「データ無効」という少なくとも５つの状態を設定しており、前記保証手段は、データが書き込まれるセクタの状態情報を「未使用」から「データ書き込み中」へ更新する手段と、該セクタへのデータの書き込みが完了した後、該セクタの状態情報を「データ書き込み中」から「データ書き込み完了」へ更新する手段と、該セクタにデータを新規に書き込んだ場合は、該セクタの状態情報を「データ書き込み完了」から「データ有効」へ更新し、該セクタに既存のデータの更新データを書き込んだ場合は、該既存のデータが書き込まれている他のセクタの状態情報を「データ有効」から「データ無効」に更新後、該更新データが書き込まれている該セクタの状態情報を「データ書き込み完了」から「データ有効」へ更新する手段とを含んでもよい。

【００４２】電力供給が回復した時のファイルシステムの初期化において、前記消去ブロックの状態情報として「元ブロック消去中」の消去ブロックが存在するか検索し、該消去ブロックが存在する場合は、該消去ブロックへのデータ転送元ブロックを、前記記憶手段内の各消去ブロックの状態情報から取得する手段と、取得したデータ転送元ブロックのデータを消去し、消去完了後、この消去ブロックの状態情報を「データ有」から「未使用」へ更新する手段と、消去ブロックの状態情報が「元ブロック消去中」の消去ブロックの状態情報を「元ブロック消去中」から「元ブロックフォーマット中」へ更新し、データを消去した消去ブロックをフォーマットして新たな予備消去ブロックとして確保する手段と、フォーマット完了後、消去ブロックの状態情報が「元ブロックフォーマット中」の消去ブロックの状態情報を「元ブロックフォーマット中」から「データ有」へ更新する手段とを含んでもよい。

【００４３】電力供給が回復した時のファイルシステムの初期化において、前記消去ブロックの状態情報として「元ブロックフォーマット中」の消去ブロックが存在するか検索し、該消去ブロックが存在する場合は、該消去ブロックへのデータ転送元ブロックを、前記記憶手段内の各消去ブロックの状態情報から取得する手段と、取得したデータ転送元ブロックをフォーマットして新たな予備消去ブロックとして確保する手段と、フォーマット完了後、消去ブロックの状態情報が「元ブロックフォーマット中」の消去ブロックの状態情報を「元ブロックフォーマット中」から「データ有」へ更新する手段とを含んでもよい。

【００４４】

【発明の実施の形態】以下、本発明の実施形態を添付図面を参照して説明する。

【００４５】図１は、本発明のファイルシステムの一実施形態を示している。図１において、１はファイルシス

テム、２はファイルシステム１の処理を制御するファイルシステム制御部、３はファイルシステム１が制御情報を格納するために使用するファイルシステムメモリ部、４はフラッシュメモリ部５へのデータ処理を制御するフラッシュメモリ制御部、５はフラッシュメモリ部、６はファイルシステム１へデータの処理を依頼するアプリケーションもしくはオペレーティングシステムである。

【００４６】図２は、図１におけるフラッシュメモリ部５に格納されているデータの構成図である。フラッシュメモリ部５の各消去ブロック５１は、データの取扱い単位である複数の物理セクタ５２に分割されて配置されており、各消去ブロック５１内で一意の物理セクタ番号を有する。各消去ブロック51のサイズが６４Ｋバイトで、各物理セクタ５２のサイズが５１２バイトであるとした場合、各消去ブロック５１には１２８個の物理セクタ５２が存在することになる。

【００４７】ここで、フラッシュメモリ部５内の複数の消去ブロック５１のうちの任意の１個の消去ブロック５１をファイルシステム１の再構築用に用いられるデータ未書き込みの予備消去ブロックとして、また各消去ブロック５１の先頭のセクタ５２をそれぞれの消去ブロックの制御情報を格納するブロックコントロールセクタ５２ａとして、予め確保しておく。

【００４８】消去ブロック５１において、ブロックコントロールセクタ５２ａには、先頭から該消去ブロック５１の状態情報としての２バイト、セクタ＃１の物理セクタ５２の状態情報としての２バイト、セクタ＃２～＃１２７の各物理セクタ５２の状態情報としてのそれぞれの２バイトが順次配列され、よって２バイト×１２８個＝２５６バイトの情報を有している。

【００４９】消去ブロック５１の状態情報は、ファイルシステム制御部２が割り当てを行う０～２５５の論理ブロック番号、及び５種類のブロック状態のいずれかを表すデータからなる。５種類のブロック状態とは、（１）「未使用」を表す１１１１１１１１ｂ（ｂは２進数を表わす。以下同様）、（２）「データ転送中」を表す１１１１１１１０ｂ、（３）「元ブロック消去中」を表す１１１１１１００ｂ、（４）「データ有」を表す１１１１１０００ｂ、（５）「ブロックフル」を表す１１１１０００ｂである。

【００５０】消去ブロック５１の各セクタ５２の状態情報は、消去ブロック５１の状態情報と同様に、ファイルシステム制御部２が割り当てを行う０～４０９５のシステム内で一意の論理セクタ番号、及び５種類のセクタ状態のいずれかを表すデータからなる。５種類のセクタ状態とは、（Ｉ）「未使用」を表す１１１１ｂ、（ＩＩ）「データ書き込み中」を表す１１１０ｂ、（ＩＩＩ）「データ書き込み完了」を表す１１００ｂ、（ＩＶ）「データ有効」を表す１０００ｂ、（Ｖ）「データ無効」を表す０００ｂである。

【００５１】図３は、図１におけるファイルシステムメモリ部３に格納されている各消去ブロック５１に関するデータの構成図（以下、ブロック情報表１０と呼称）である。

【００５２】図３において、物理ブロック番号、論理ブロック番号、及びブロック状態を１つの情報単位としており、例えば物理ブロック番号＝０、論理ブロック番号＝０、ブロック状態＝データ有となっている消去ブロック５１、物理ブロック番号＝１、論理ブロック番号＝１、ブロック状態＝ブロックフルとなっている消去ブロック５１等が存在することを示している。

【００５３】図４は、図１におけるファイルシステムメモリ部３に格納されている各セクタ５２に関するデータの構成図（以下、セクタ情報表１１と呼称）である。

【００５４】図４において、物理ブロック番号、物理セクタ番号、論理セクタ番号、及びセクタ状態を１つの情報単位としており、例えば物理ブロック番号＝０、物理セクタ番号＝０、論理セクタ番号＝１００、セクタ状態＝データ有効となっているセクタ５２、物理ブロック番号＝０、物理セクタ番号＝１、論理セクタ番号＝１０、セクタ状態＝データ無効となっているセクタ５２等が存在することを示している。

【００５５】図５は、図１におけるファイルシステムメモリ部３に格納されている各消去ブロック５１毎に、各セクタ５２の数をそれぞれのセクタ状態に応じて整理したデータの構成図（以下、セクタ情報表１２と呼称）である。

【００５６】図５において、物理ブロック番号、未使用セクタ数、データ有効セクタ数、及びデータ無効セクタ数を１つの情報単位としており、例えば物理ブロック番号＝０、未使用セクタ数＝１００、データ有効セクタ数＝２０、データ無効セクタ数＝７となっている消去ブロック５１、物理ブロック番号＝１、未使用セクタ数＝０、データ有効セクタ数＝５０、データ無効セクタ数＝７７となっている消去ブロック５１等が存在することを示している。

【００５７】図３のブロック情報表１０、図４のセクタ情報表１１、及び図５のセクタ情報表１２は、ファイルシステム１が起動する段階において、フラッシュメモリ部５から各消去ブロック５１のブロックコントロールセクタ５２ａを読み出すことによってそれぞれ作成される。

【００５８】ブロック情報表１０、セクタ情報表１１、及びセクタ情報表１２を作成しない場合においても、これらの表の情報が各消去ブロック５１のブロックコントロールセクタ５２ａに格納されているため、ファイルシステム１を使用することは可能である。しかしながら、ファイルシステムメモリ部３の動作速度がフラッシュメモリ部５よりも高速なため、ファイルシステムメモリ部３にブロック情報表１０、セクタ情報表１１、及びセク

タ情報表１２を作成して置いておく方が、フラッシュメモリ部５の内部状態やデータの格納場所を高速に把握することができる。また、フラッシュメモリ部５とのデータのやり取りが減るので、ファイルシステムメモリ部３を利用した方が、ファイルシステム１の動作速度を高速にする上で非常に有効である。

【００５９】図６は、本実施形態におけるフラッシュメモリ部５に格納されたデータの読み出し処理に関するフローチャートである。

【００６０】まず、アプリケーションもしくはオペレーティングシステム６においてフラッシュメモリ部５からのデータの読み出し要求が発生した場合、この読み出し要求と共に論理セクタ番号がアプリケーションもしくはオペレーティングシステム６からファイルシステム制御部２に与えられる。ファイルシステム制御部２は、ファイルシステムメモリ部３内のセクタ情報表１１をアクセスして、与えられた論理セクタ番号を検索し、この論理セクタ番号に対応するセクタ状態が「データ有効」となっていることを確認し、この論理セクタ番号に対応する物理セクタ番号を取得し、この物理セクタ番号をフラッシュメモリ制御部４に与える（ステップ１２１）。フラッシュメモリ制御部４は、この物理セクタ番号に基いてフラッシュメモリ部５をアクセスし、この物理セクタ番号に該当する物理的な格納場所のセクタに対して読み出しを行い、読み出したデータをファイルシステム制御部２に与える（ステップ１２２）。読み出すべき残りのデータがある場合は、ステップ１２１へ戻り（ステップ１２３，Ｙｅｓ）、読み出すべき残りのデータがある場合は、終了となる（ステップ１２３，Ｎｏ）。

【００６１】図７Ａ～図７Ｃは、本実施形態におけるフラッシュメモリ部５へのデータの書き込み処理に関するフローチャートである。

【００６２】まず、アプリケーションもしくはオペレーティングシステム６においてフラッシュメモリ部５へのデータの書き込み要求が発生した場合、この書き込み要求と共に論理セクタ番号及び書き込むデータがアプリケーションもしくはオペレーティングシステム６からファイルシステム制御部２に与えられる。ファイルシステム制御部２は、ファイルシステムメモリ部２内のセクタ情報表１２をアクセスして、未使用セクタ数が最も多い物理ブロック番号（書き込み可能な消去ブロック５１を示す）を取得し（ステップ１３１）、取得できたかどうかの確認を行い（ステップ１３２）、この物理ブロック番号が取得出来なかった場合は（ステップ１３２，Ｎｏ）、書き込み可能な消去ブロック５１を確保するために後述するファイルシステム１の再構築を行う（ステップ１３３）。

【００６３】また、未使用セクタ数が最も多い物理ブロック番号を取得できれば（ステップ１３２，Ｙｅｓ）、ファイルシステム制御部２は、セクタ情報表１１をアクセスして、ステップ１３１又は１３３において取得した物理ブロック番号に対応する各物理セクタ番号のうちからセクタ状態が「未使用」となっている物理セクタ番号を取得し、更にセクタ情報表１２をアクセスして、該物理ブロック番号に対応する未使用セクタ数を－１とし、データ有効セクタ数を＋１とする（ステップ１３４）。

【００６４】この後、ファイルシステム制御部２は、ブロック情報表１０をアクセスして、ステップ１３１又は１３３において取得した物理ブロック番号に対応するブロック状態が「未使用」であるかの確認を行い（ステップ１３５）、ブロック状態が「未使用」であれば（ステップ１３５，Ｙｅｓ）、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、該物理ブロック番号に対応する消去ブロック５１のブロックコントロールセクタ５２ａの論理ブロック番号を該物理ブロック番号に更新し、ブロックコントロールセクタ５２ａのブロック状態を「未使用」から「データ有」へ更新する。同様に、ブロック情報表１０をアクセスし、ステップ１３１又は１３３において取得した物理ブロック番号に対応する論理ブロック番号を該物理ブロック番号に更新し、ブロック状態を「未使用」から「データ有」へ更新し（ステップ１３６）、ステップ１３７に移る。ブロック状態が「未使用」でなければ（ステップ１３５，Ｎｏ）、ステップ１３６をジャンプしてステップ１３７に移る。

【００６５】次に、ファイルシステム制御部２は、セクタ情報表１２をアクセスし、ステップ１３１又は１３３において取得した物理ブロック番号に対応する未使用セクタ数が０かの確認を行い（ステップ１３７）、未使用セクタ数が０であれば（ステップ１３７，Ｙｅｓ）、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、該物理ブロック番号に対応する消去ブロック５１のブロックコントロールセクタ５２ａのブロック状態を「データ有」から「ブロックフル」へ更新する。同様に、ブロック情報表１０をアクセスし、ステップ１３１又は１３３において取得した物理ブロック番号に対応するブロック状態も「データ有」から「ブロックフル」へ更新する（ステップ１３８）。

【００６６】次に、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、ステップ１３１又は１３３において取得した物理ブロック番号並びにステップ１３４において取得した物理セクタ番号に対応する消去ブロック５１のブロックコントロールセクタ５２ａの論理セクタ番号にファイルシステム制御部２に与えられた論理セクタ番号を書き込み、このセクタ５２のセクタ状態を「未使用」から「データ書き込み中」へ更新する。同様に、セクタ情報表１１をアクセスし、ステップ１３１又は１３３において取得した物理ブロック番号並びにステップ１３４において取得した物理セクタ番号に対応する論理セクタ番号

をファイルシステム制御部２に与えられた論理セクタ番号に更新し、セクタ状態を「未使用」から「データ書き込み中」へ更新する（ステップ１３９）。

【００６７】この後、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、ステップ１３１又は１３３において取得した物理ブロック番号並びにステップ１３４において取得し物理セクタ番号に対応するセクタ５２に、ファイルシステム制御部２に与えられたデータを書き込む(ステップ１４０）。

【００６８】ファイルシステム制御部２は、フラッシュメモリ部５内のセクタ５２への書き込み終了後、ステップ１３１又は１３３において取得した物理ブロック番号並びにステップ１３４において取得した物理セクタ番号に対応する消去ブロック５１のブロックコントロールセクタ５２ａのセクタ状態を「データ書き込み中」から「データ書き込み完了」へ更新する。同様に、セクタ情報表１１をアクセスし、ステップ１３１又は１３３において取得した物理ブロック番号並びにステップ１３４において取得した物理セクタ番号に対応するセクタ状態を「データ書き込み中」から「データ書き込み完了」へ更新する（ステップ１４１）。

【００６９】そして、ファイルシステム制御部２は、フラッシュメモリ部５内のセクタ５２に書き込んだデータが既にフラッシュメモリ部５に書き込まれていたデータの更新データであるかの確認を行う（ステップ１４２）。ここで、既存データの更新であれば（ステップ１４２，Ｙｅｓ）、ファイルシステム制御部２は、セクタ情報表１１をアクセスして、ステップ１４０において書き込みを実行したセクタ５２の論理セクタ番号と同じ論理セクタ番号を有しかつステップ１４０において書き込みを実行したセクタ５２の物理セクタ番号とは異なる他のセクタ５２の物理セクタ番号及び物理ブロック番号、つまり既存のデータが格納されたセクタ５２の物理セクタ番号及び物理ブロック番号を取得する（ステップ１４３）。そして、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、これらの物理ブロック番号並びに物理セクタ番号に対応する消去ブロック５１のブロックコントロールセクタ５２ａのセクタ状態を「データ有効」から「データ無効」へ更新する。同様に、セクタ情報表１１をアクセスし、これらの物理ブロック番号並びに物理セクタ番号に対応するセクタ状態を「データ有効」から「データ無効」へ更新する。更に、セクタ情報表１２をアクセスし、この物理ブロック番号に対応するデータ有効セクタ数を－１とし、データ無効セクタ数を＋１する（ステップ１４４）。

【００７０】この後、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、ステップ１３１又は１３３において取得した物理ブロック番号並びにステップ１３４において取得した物理セクタ番号に対応する消去ブロック５１のブロックコントロールセクタ５２ａのセクタ状態を「データ書き込み完了」から「データ有効」へ更新する。同様に、セクタ情報表１１をアクセスし、これらの物理ブロック番号並びに物理セクタ番号に対応するセクタ状態も「データ書き込み完了」から「データ有効」へ更新する（ステップ１４５）。

【００７１】引き続いて、ファイルシステム制御部２は、書き込みデータがまだ残っているかの確認を行い（ステップ１４６）、書き込みデータがまだ残っている場合は（ステップ１４６，Ｙｅｓ）、ステップ１３１へ戻り、書き込みデータがなくなれば（ステップ１４６，Ｎｏ）、終了する。

【００７２】図８Ａ～図８Ｂは、本実施形態におけるファイルシステム１の再構築処理に関するフローチャートである。

【００７３】まず、図７Ａ～図７Ｃのフローチャートで説明した様に書き込み可能な消去ブロック５１が取得出来なかった場合は（ステップ１３２，Ｎｏ）、書き込み可能な消去ブロック５１を確保するためにファイルシステムの再構築を行う（ステップ１３３）。このステップ１３３の処理が図８のフローチャートに示す処理に該当する。

【００７４】図８Ａ～図８Ｂにおいて、ファイルシステム制御部２は、セクタ情報表１２をアクセスし、データ無効セクタ数が最も多い物理ブロック番号を取得し、更にブロック情報表１０をアクセスし、取得した物理ブロック番号に対応する論理ブロック番号を取得する（ステップ１５１）。そして、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、ファイルシステムの再構築処理用に予め未使用状態で確保しておいた予備消去ブロック５１のブロックコントロールセクタ５２ａの論理ブロック番号をステップ１５１において取得した論理ブロック番号へ更新し、このブロックコントロールセクタ５２ａのブロック状態を「未使用」から「データ転送中」へ更新する。同様に、ブロック情報表１０をアクセスし、予備消去ブロック５１の論理ブロック番号をステップ１５１において取得した論理ブロック番号へ更新し、ブロック状態を「未使用」から「データ転送中」へ更新する（ステップ１５２）。

【００７５】次に、ファイルシステム制御部２は、データをステップ１５１において取得した物理ブロック番号の消去ブロック５１から予備消去ブロック５１へ効率良くコピーするにために、セクタ情報表１１をアクセスし、ステップ１５１において取得した物理ブロック番号に対応する各物理セクタ番号のうちからセクタ状態が「データ有効」である物理セクタ番号を検索して取得する（ステップ１５３）。セクタ状態が「データ有効」の

セクタ５２があれば（ステップ１５４，Ｙｅｓ）、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、ステップ１５１において取得した物理ブロック番号並びにステップ１５３において取得した物理セクタ番号に対応する消去ブロック５１のブロックコントロールセクタ５２ａのセクタ状態「データ有効」を予備消去ブロック５１の同じ物理セクタ番号に対応するブロックコントロールセクタ５２ａのセクタ状態にコピーする。同様に、ファイルシステム制御部２は、セクタ情報表１１をアクセスし、ステップ１５１において取得した物理ブロック番号並びにステップ１５３において取得した物理セクタ番号のセクタ状態「データ有効」を予備消去ブロック５１の同じ物理セクタ番号のセクタ状態にコピーする。同時に、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、ステップ１５１において取得した物理ブロック番号並びにステップ１５３において取得した物理セクタ番号に対応する消去ブロック５１のセクタ５２から予備消去ブロック５１の同じ物理セクタ番号のセクタ５２へとデータをコピーする（ステップ１５５）。

【００７６】ファイルシステム制御部２は、ステップ１５１において取得した物理ブロック番号の消去ブロック５１に未処理のセクタ５２があるかの確認を行う（ステップ１５６）。未処理のセクタ５２がある場合は、ステップ１５３へ戻る。未処理のセクタ５２がない場合は、データのコピーが完了したため、ファイルシステム制御部２は、セクタ情報表１２をアクセスし、ステップ１５１において取得した物理ブロック番号に対応するデータ有効セクタ数を予備消去ブロック５１のデータ有効セクタ数へコピーし、またデータ無効セクタ数を０とし、データ有効セクタ数とデータ無効セクタ数と未使用セクタ数の和が総セクタ数（１２７）となる様に、未使用セクタ数を計算して設定する（ステップ１５７）。

【００７７】次に、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、予備消去ブロック５１のブロックコントロールセクタ５２ａのブロック状態を「データ転送中」から「元ブロック消去中」へ更新し（ステップ１５８）、ステップ１５１において取得した物理ブロック番号の消去ブロック５１のデータを消去し（ステップ１５９）、予備消去ブロック５１のブロックコントロールセクタ５２ａのブロック状態を「元ブロック消去中」から「データ有」へ更新する。同様に、ファイルシステム制御部２は、ブロック情報表１０をアクセスし、予備消去ブロック５１のブロック状態を「データ転送中」から「元ブロック消去中」へ更新し（ステップ１５８）、ステップ１５９を経た後に予備消去ブロック５１のブロック状態を「元ブロック消去中」から「データ有」へ更新する。これによって、ステップ１５１において取得した物理ブロック番号のブロック、つまりデータを消去したブロックが新しい予備消去ブロック５１として確保される（ステップ１６０）。

【００７８】ここで、ステップ１５９においてデータを消去した消去ブロック５１のブロックコントロールセクタ５２ａのブロック状態は、データ消去により自動的に「データ有」あるいは「ブロックフル」から「未使用」となる。また、ブロック情報表１０、セクタ情報表１１及びセクタ情報表１２では、ステップ１５９においてデータを消去した消去ブロック５１が予備消去ブロックとなり、フラッシュメモリ部５へのデータの書き込みにおいて予備消去ブロックが参照されることはないため、この予備消去ブロックの物理ブロック番号に対応する項目の更新は不要である。

【００７９】なお、本実施例においては、ファイルシステム１の再構築の処理は、消去ブロック１つを消去する場合を示したが、ステップ１５１からの一連の処理を複数回実行することも可能である。

【００８０】図９Ａ～図９Ｄは、本実施形態におけるファイルシステム１の初期化に関するフローチャートである。この初期化は、ファイルシステム１の立ち上げ時、つまり電力供給が一旦遮断された後の電力供給再開時に一度実行され、フラッシュメモリ部５内のデータの読み出しを保証するものである。

【００８１】まず、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、各消去ブロック５１のブロックコントロールセクタ５２ａのブロック状態を読み出し、この読み出したブロック情報に基いて、ブロック情報表１０のブロック情報を更新し（ステップ１７１）、残りの消去ブロック５１が存在するかの確認を行い（ステップ１７２）、残りの消去ブロック５１が無くなるまで、ステップ１７１の処理を繰り返す。

【００８２】次に、ファイルシステム制御部２は、ステップ１７１において更新したブロック情報表１０をアクセスし、ブロック状態「データ転送中」が存在するかの確認を行い（ステップ１７３）、ブロック状態「データ転送中」が存在する場合は、このブロック状態の物理ブロック番号を取得し、この物理ブロック番号に対応するブロック状態を「データ転送中」から「未使用」へ更新する（ステップ１７４）。同時に、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、この物理ブロック番号のブロックのデータを消去する。

【００８３】ここで、図８Ａ～図８Ｂのステップ１５１～１５７の途中で電力供給が遮断され、予備消去ブロック５１へのデータ転送が中断した場合は、この予備消去ブロック５１のブロック状態が「データ転送中」として保持される。この後に電力供給が再開されたときには、ステップ１７３、１７４の処理において、データ転送が

中断したブロック状態「データ転送中」の予備消去ブロック５１のデータが消去され、この予備消去ブロック５１のブロック状態が「未使用」となる。これによって不完全なデータが予備消去ブロック５１から読み出されることが防止される。

【００８４】次に、ファイルシステム制御部２は、ステップ１７１において更新したブロック情報表１０をアクセスし、ブロック状態が「元ブロック消去中」であるブロックが存在するかの確認を行い（ステップ１７５）、ブロック状態が「元ブロック消去中」であるブロックが存在する場合は、このブロックの論理ブロック番号と同じ論理ブロック番号であってかつブロック状態が「データ有」である物理ブロック番号を取得する。そして、ファイルシステム制御部２は、ブロック情報表１０をアクセスし、この物理ブロック番号に対応するブロック状態を「未使用」へ更新する。同時に、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、この物理ブロック番号のブロックのデータを消去する。

【００８５】また、ファイルシステム制御部２は、ステップ１７５において取得したブロック状態が「元ブロック消去中」である物理ブロック番号の消去ブロック５１のブロックコントロールセクタ５２ａのブロック状態情報を「元ブロック消去中」から「データ有」へ更新し、ブロック情報表１０における同じ物理ブロック番号に対応するブロック状態を「データ有」へ更新する（ステップ１７６）。

【００８６】この後、ファイルシステム制御部２は、消去ブロック５１のブロックコントロールセクタ５２ａのセクタ状態を読み出し、このセクタ情報に基いてファイルシステムメモリ部３内のセクタ情報表１１を更新し（ステップ１７７）、残りのセクタ情報が存在するかの確認を行い（ステップ１７８）、残りのセクタ情報が存在する場合は、ステップ１７７へ戻り、残りのセクタ５２がなくなるまで繰り返し、同様に残りの消去ブロック５１が存在する場合は、ステップ１７７へ戻り、残りの消去ブロック５１がなくなるまで繰り返す。

【００８７】ここで、図８Ａ～図８Ｂのステップ１５８～１６０の途中で電力供給が遮断された場合は、この予備消去ブロック５１のブロック状態が「元ブロック消去中」として保持されており、この予備消去ブロック５１へのデータ転送が終了している。この後に電力供給が再開されると、ステップ１７５～１７８の処理において、予備消去ブロック５１のデータを保持し、データ転送元の消去ブロック５１のデータを消去している。これによってデータ転送元の消去ブロック５１から不完全なデータが読み出されることが防止される。

【００８８】次に、ファイルシステム制御部２は、ステップ１７７において更新したセクタ情報表１１をアクセスし、セクタ状態が「データ書き込み中」であるセクタ５２が存在するかの確認を行い（ステップ１７９）、セクタ状態が「データ書き込み中」であるセクタ５２が存在する場合は、ステップ１７９において取得したセクタ状態「データ書き込み中」に対応する物理ブロック番号及び物理セクタ番号を取得する。そして、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、これらの物理ブロック番号及び物理セクタ番号に対応する消去ブロック５１のブロックコントロールセクタ５２ａのセクタ状態を「データ書き込み中」から「データ無効」へ更新する。同様に、ファイルシステム制御部２は、セクタ情報表１１をアクセスし、これらの物理ブロック番号及び物理セクタ番号に対応するセクタ状態を「データ書き込み中」から「データ無効」へ更新する（ステップ１８０）。

【００８９】次に、ファイルシステム制御部２は、セクタ情報表１１をアクセスし、セクタ状態が「データ書き込み完了」であるセクタ５２が存在するかの確認を行い（ステップ１８１）、セクタ状態が「データ書き込み完了」であるセクタ５２が存在する場合は、このセクタ５２の論理セクタ番号と同じ論理セクタ番号を有しかつセクタ状態が「データ有効」のセクタ５２が存在するかの確認を行い（ステップ１８２）、このセクタ５２が存在する場合は、このセクタ５２の物理ブロック番号及び物理セクタ番号を取得する。そして、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、これらの物理ブロック番号及び物理セクタ番号に対応する消去ブロック５１のブロックコントロールセクタ５２ａのセクタ状態を「データ有効」から「データ無効」へ更新する。同様に、ファイルシステム制御部２は、セクタ情報表１１をアクセスし、これらの物理ブロック番号及び物理セクタ番号に対応するセクタ状態を「データ有効」から「データ無効」へ更新する（ステップ１８３）。

【００９０】次に、ファイルシステム制御部２は、セクタ情報表１１をアクセスし、ステップ１８１において取得したセクタ状態が「データ書き込み完了」であるセクタ５２の物理ブロック番号及び物理セクタ番号を取得する。そして、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、これらの物理ブロック及び物理セクタ番号に対応する消去ブロック５１のブロックコントロールセクタ５２ａののセクタ状態を「データ書き込み完了」から「データ有効」へ更新する。同様に、ファイルシステム制御部２は、セクタ情報表１１をアクセスし、これらの物理ブロック番号及び物理セクタ番号に対応するセクタ状態を「データ書き込み完了」から「データ有効」へ更新する（ステップ１８４）。

【００９１】ここで、図７Ａ～図７ｃのステップ１３９～１４１の途中で電力供給が遮断され、データの書き込みが中断した場合は、セクタ状態が「データ書き込み

中」として保持され、またデータの書き込みが完了した場合は、セクタ状態が「データ書き込み完了」として保持される。この後に電力供給が再開されると、ステップ１７９～１８４の処理において、データの書き込みが中断したセクタ状態「データ書き込み中」のセクタ５２のデータが消去され、このセクタ５２のセクタ状態が「データ無効」となる。これによってデータの書き込みが中断したセクタ５２から不完全なデータが読み出されることが防止される。また、セクタ状態「データ書き込み完了」のセクタ５２のデータが保持され、このセクタ状態が「データ有効」に更新されるので、このセクタ５２からデータを正しく読み出すことができる。

【００９２】次に、ファイルシステム制御部２は、セクタ情報表１１をアクセスし、物理ブロック番号毎に、同一物理ブロック番号に対応するセクタ状態「未使用」の数、セクタ状態「データ有効」の数、セクタ状態「データ無効」の数をそれぞれ計数し、これらの計数値に基いてファイルシステムメモリ部３内のセクタ情報表１２の物理ブロック番号、未使用セクタ数、データ有効セクタ数及びデータ無効セクタ数を更新し（ステップ１８５）、残りの物理ブロック番号が存在するかの確認を行い（ステップ１８６）、残りの物理ブロック番号が存在する場合は、ステップ１８５へ戻り、残りの物理ブロック番号がなくなるま繰り返す。

【００９３】この後、ファイルシステム制御部２は、ステップ１７１、ステップ１７４、ステップ１７５において更新されたブロック情報表１０に基いて、ブロック状態が「未使用」である物理ブロック番号に対応する１つの消去ブロック５１を予備消去ブロック５１として確保する（ステップ１８７）。

【００９４】この様に本実施形態においては、データの書き込みやファイルシステムの再構築に際し、各消去ブロック５１や各セクタ５２の状態情報をきめ細かく書き換えているため、電力供給が不用意に遮断されも、この後に電力供給が再開されたときには、各消去ブロック５１や各セクタ５２の状態情報に基いて、消去ブロック５１やセクタ５２の不確定なデータを消去して、正しいデータのみを保存することができる。

【００９５】上記実施形態においては、現在のブロック状態として、（１）「未使用」を表す１１１１１１１１ｂ（ｂは２進数を表わす。以下同様）、（２）「データ転送中」を表す１１１１１１１０ｂ、（３）「元ブロック消去中」を表す１１１１１１００ｂ、（４）「データ有」を表す１１１１０００ｂ、（５）「ブロックフル」を表す１１１００００ｂという５種類のブロック状態を例示したが、１と０の組み合わせを複雑にしたり、ビット数を増やしても良く、これによって各状態の違いをより確実に判定することが可能になる。

【００９６】また、よりきめ細かくブロック状態を管理するため、ブロックの状態数を６つ以上に設定しても良く、これによってよりきめ細かく状態を管理することが可能になる。

【００９７】また、上記実施形態においては、現在のセクタ状態として、（Ｉ）「未使用」を表す１１１１ｂ、（ＩＩ）「データ書き込み中」を表す１１１０ｂ、（ＩＩＩ）「データ書き込み完了」を表す１１００ｂ、（ＩＶ）「データ有効」を表す１０００ｂ、（Ｖ）「データ無効」を表す０００ｂという５種類のセクタ状態を例示したが、１と０の組み合わせを複雑にしたり、ビット数を増やしても良く、これによって各状態の違いをより確実に判定することが可能になる。

【００９８】また、よりきめ細かくセクタ状態を管理するため、セクタ５２の状態数を６つ以上に設定しても良く、これによってよりきめ細かく状態を管理することが可能になる。

【００９９】上記実施形態においては、均等ブロックサイズであるフラッシュメモリを前提として説明したが、ブートブロック構成のフラッシュメモリにおいても、ブートブロック（例えば８Ｋバイトのサイズ）用の任意の１個の消去ブロック（８Ｋバイト）と、それ以外のブロック（例えば６４Ｋバイトのサイズ）用の任意の１個の消去ブロック（６４Ｋバイト）をファイルシステム１の再構築用の予備ブロックとして、予め確保しても良く、これによって均等ブロックあるいはブートブロック構成のフラッシュメモリのうちからシステムの目的に合うフラッシュメモリあるいはより安価なフラッシュメモリを採用することが可能となる。また、ブートブロック（例えば８Ｋバイト）用の予備ブロックとして、８Ｋバイトより大きいサイズのブロック（例えば６４Ｋバイト）を予備ブロックに割り当てることも可能であり、この場合は、予備ブロックのサイズを一定にできるため、管理が簡単になる。

【０１００】本実施形態のファイルシステム１が複数個のフラッシュメモリを備える場合は、１個のフラッシュメモリ毎に任意の１個の消去ブロックをファイルシステムの再構築用の予備ブロックとして予め確保しておくのではなく、複数個のフラッシュメモリに対し１個の予備ブロックを確保したり、複数個のフラッシュメモリに含まれるブロックサイズが異なる各消去ブロック毎に、少なくとも１個の予備ブロックを確保しても良く、この場合は、確保する予備ブロックの数を少なくすることができる。

【０１０１】図９Ｂに示すステップ１７６においては、ファイルシステム制御部２は、ステップ１７５において取得したブロック状態「元ブロック消去中」の消去ブロック５１の論理ブロック番号と同じ論理ブロック番号であって、且つブロック状態が「データ有」である物理ブロック番号を取得する。そして、ファイルシステム制御部２は、ブロック情報表１０をアクセスし、この物理ブロック番号に対応するブロック状態を「未使用」へ更新

SAMSUNG EX1002, Page 412 of 1031

する。同時に、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、この物理ブロック番号の消去ブロック５１のデータを消去する。また、ファイルシステム制御部２は、ステップ１７５において取得したブロック状態「元ブロック消去中」の物理ブロック番号の消去ブロック５１のブロックコントロールセクク５２ａのブロック状態を「元ブロック消去中」から「データ有」へ更新し、ブロック情報表１０における同じ物理ブロック番号に対応するブロック状態を「データ有」へ更新する。

【０１０２】ここで、消去ブロック５１についての複数の状態情報としては、『未使用」、「データ有」、「ブロックフル」、「データ転送中」、「元ブロック消去中」の５つに加え、「元ブロック消去完了」を設定した場合の例として以下のような処理の流れにすることが可能である。

【０１０３】図１３は、図９Ａ～図９Ｄのファイルシステム１の初期化に関するフローチャートの内、本実施形態におけるステップ１７４からステップ１７７の間に相当する部分を抜き出したフローチャートを示す図である。

【０１０４】まず、ファイルシステム制御部２は、ステップ１７１において更新したブロック情報表１０をアクセスし、ブロック状態「元ブロック消去完了」が存在するかを確認し（ステップ１９１）、ブロック状態「元ブロック消去完了」が存在する場合は、処理はステツプ１９２およびステップ１９３をスキップしてステップ１９４へジャンプする。

【０１０５】次に、ファイルシステム制御部２は、ステップ１７１において更新したブロック情報表１０をアクセスし、ブロック状態「元ブロック消去中」が存在するかを確認し（ステップ１９２）、ブロック状態「元ブロック消去中」が存在する場合は、この消去ブロック５１の論理ブロック番号と同じ論理ブロック番号であって、且つブロック状態が「データ有」である物理ブロック番号を取得し、ファイルシステム制御部２は、フラッシュメモリ制御部４を通じてフラッシュメモリ部５をアクセスし、この物理ブロック番号の消去ブロック５１のデータを消去する。

【０１０６】この後、ファイルシステム制御部２は、ステツプ１９２において取得したブロック状態が「元ブロック消去中」である物理ブロック番号の消去ブロック５１のブロックコントロールセクタ５２ａのブロック状態を「元ブロック消去中」から「元ブロック消去完了」へ更新する。

【０１０７】また、ファイルシステム制御部２は、ブロック情報表１０をアクセスし、データ消去をした消去ブロック５１の物理ブロック番号に対応するブロック状態を「未使用」へと更新する（ステップ１９３）。

【０１０８】次に、ブロック状態「元ブロック消去完了」へ更新した物理ブロック番号の消去ブロック５１のブロックコントロールセクタ５２ａのブロック状態を「元ブロック消去完了」から「データ有」へ更新し、ブロック情報表１０におけるこの物理ブロック番号に対応するブロック状態を「元ブロック消去中」から「データ有」へ更新する（ステップ１９４）。

【０１０９】この様に、ブロック状態「元ブロック消去完了」を追加することにより、比較的処理時間の長いブロック消去処理をスキップすることが可能、すなわち、電力供給の遮断直前のブロック状態が「元ブロック消去中」から「元ブロック消去完了」へ更新された直後であった場合は、電力供給が回復した時の初期化処理時間を短縮することが可能である。

【０１１０】他の実施形態では、図１２に示すように、消去ブロック５１についての複数の状態情報として、「未使用」、「データ有」、「データ転送中」、「元ブロック消去中」、「元ブロックフォーマット中」の５つを設定しても良い。

【０１１１】この場合、再構築用に予め確保されている予備消去ブロック５１の状態情報を「未使用」から「データ転送中」へ更新する手段と、データ転送元の消去ブロック５１内のデータを該予備消去ブロック５１へ転送して記憶させる手段と、該データの転送並びに記憶の完了後、該予備消去ブロック５１のブロック状態情報を「データ転送中」から「元ブロック消去中」へ更新し、該データ転送元の消去ブロック５１のデータを消去し、このデータ転送元の消去ブロック５１を新たな予備消去ブロック５１として確保し、この新たな予備消去ブロック５１の状態情報を「データ有」から「未使用」へ更新する手段と、データ転送先の該予備消去ブロック５１の状態情報を「元ブロック消去中」から「元ブロックフォーマット中」へ更新し、データ転送元の新たな予備消去ブロック５１へ新たな予備消去ブロック５１の消去可能回数などを書き込む（フォーマットする）手段と、フォーマット完了後、データ転送先の該予備消去ブロック５１の状態情報を「元ブロックフォーマット中」から「データ有」へ更新する手段とを合む構成が考えられる。

【０１１２】この様に、消去ブロック５１の状態情報として「未使用」、「データ有」、「データ転送中」、「元ブロック消去中」、「元ブロックフォーマット中」という５つの状態を利用した場合、消去ブロックの状態を正確に把握してきめ細かにフラッシュメモリを管理することができ、フラッシュメモリに格納されている既存のデータを破壊することなく、また既存のデータに矛盾を生じさせることなく、電力供給の遮断直前に実行中であった前段階へファイルシステムを自動的に回復させることが可能となる。

【０１１３】他の実施形態における電力回復時のファイルシステム１の初期化においてブロック状態「元ブロック消去中」の消去ブロック５１が検出された場合は、デ

ータ転送元の消去ブロック５１のデータを消去する。または、更に既存データの正確性を期すために、該初期化において「元ブロック消去中」だけでなく「元ブロックフォーマット中」の消去ブロック５１が検出された場合もデータ転送元の消去ブロック５１のデータを消去する方法が考えられる。

【０１１４】また、消去ブロック５１の状態情報として「ブロックフル」という状態を設定する前記実施形態と異なり、ブロック状態「ブロックフル」を設定しない方法は、消去ブロックが不均等なサイズのセクタに分割されて使用される場合に特に有効である。なぜならば、不均等なサイズのデータを同一の消去ブロック内の連続したセクタに書き込む場合において、次に書き込むべきデータのサイズが不明である場合に、消去ブロックの状態情報が「ブロックフル」となるかどうかの判断が難しいためである。

【０１１５】また、各消去ブロック５１のブロック消去回数を平均化するために、各消去ブロック５１内に該消去ブロックの消去可能回数を記憶する方法がある。

【０１１６】図８Ａおよび図８Ｂに示す他の実施形態としては、ファイルシステム１の再構築時において、ファイルシステム制御部２は、データの転送元となる消去ブロックとして予備消去ブロック５１を除いた各消去ブロック５１の消去可能回数を比較し、該回数の最大値と最小値の差が一定値を越えた場合には、該回数が最大の消去ブロック５１を選択し、差が一定値を越えていない場合は、消去ブロック５１内のセクタ情報表１２をアクセスし、データ無効セクタ数が最大の消去ブロック５１を選択する。

【０１１７】この場合、各消去ブロック５１の消去可能回数のバラツキを一定の回数内に抑え、フラッシュメモリとしての書き替え回数を増やすことが可能になる。

【０１１８】また、消去ブロック５１の消去可能回数は、０から1つずつ増加させ、消去可能回数の仕様値の最大値までインクリメントする方法（この場合は消去可能回数でなく消去回数を表す）と、これとは逆に消去可能回数の仕様値の最大値から１つずつ減少させ、０までデクリメントする方法が考えられるが、後者の方法においては、消去可能回数が０になった時点で該消去ブロック５１の寿命が来たと判定できるため、消去可能回数の判定アルゴリズムを簡単にできる。

【０１１９】他の実施形態における消去ブロック５１のフォーマット手段では、フォーマットした日時を記憶することも可能であり、この場合、一定時間経過後に消去ブロック５１内のデータのリフレッシュ動作を行うことが可能になる。

【０１２０】

【発明の効果】本発明のファイルシステムは、複数のセクタをそれぞれ含む複数の消去ブロックを有する不揮発性半導体記憶装置を用いたファイルシステムにおいて、該各消去ブロック毎に、消去ブロックについての複数の状態情報のいずれかを格納すると共に、該各セクタ毎に、セクタについての複数の状態情報のいずれかを格納する記憶手段と、該不揮発性半導体記憶装置のアクセス時に、該記憶手段内の該各消去ブロックの状態情報及び該各セクタの状態情報に基いて、該不揮発性半導体記憶装置に既に格納されているデータを保証する保証手段とを具備する。

【０１２１】本発明によれば、ブロックについての複数の状態情報及びセクタについての複数の状態情報を記憶手段に格納している。このため、ファイルシステムが稼働中において、意図しない電力供給の遮断が生じても、電力供給が回復したときには、起動時の初期化処理に際し、ブロックについての複数の状態情報及びセクタについての複数の状態情報に基いて、ブロック及びセクタの状態を正確に把握してきめ細かにフラッシュメモリを管理することができ、フラッシュメモリに格納されている既存のデータを破壊することなく、また既存のデータに矛盾を生じさせることなく、電力供給の遮断直前に実行中であった前段階へファイルシステムを自動的に回復させることが可能となる。

【０１２２】一実施形態では、前記記憶手段は、データの更新に制約の無いＲＡＭ上に設けられ、前記保証手段は、前記不揮発性半導体記憶装置へのアクセス時に、該ＲＡＭ上の前記各消去ブロックの状態情報及び前記各セクタの状態情報を参照する。

【０１２３】この様にＲＡＭを利用すると、データの更新に制約がなくなり処理速度が早くなる。

【０１２４】一実施形態では、前記消去ブロックについての複数の状態情報として、「未使用」、「データ有」、「ブロックフル」、「データ転送中」、「元ブロック消去中」という少なくとも５つの状態を設定しており、前記保証手段は、再構築用に予め確保されている予備ブロックの状態情報を「未使用」から「データ転送中」へ更新する手段と、データ転送元の消去ブロック内のデータを該予備ブロックへ転送して記憶させる手段と、該データの転送並びに記憶の完了後、該予備ブロックのブロック状態情報を「データ転送中」から「元ブロック消去中」へ更新し、該データ転送元の消去ブロックのデータを消去し、このデータ転送元の消去ブロックを新たな予備ブロックとして確保し、この新たな予備ブロックの状態情報を「データ有」あるいは「ブロックフル」から「未使用」へ更新する手段と、データ転送先の該予備ブロックの状態情報を「元ブロック消去中」から「データ有」へ更新する手段とを含む。

【０１２５】この様に消去ブロックの状態情報として「未使用」、「データ有」、「ブロックフル」、「データ転送中」、「元ブロック消去中」という５つの状態を利用した場合、ブロックの状態を正確に把握してきめ細かにフラッシュメモリを管理することができ、フラッシ

ュメモリに格納されている既存のデータを破壊することなく、また既存のデータに矛盾を生じさせることなく、電力供給の遮断直前に実行中であった前段階へファイルシステムを自動的に回復させることが可能となる。

【０１２６】一実施形態では、再構築用に予め確保されている予備ブロックは、１個の不揮発性半導体記憶装置に含まれるブロックサイズが異なる各消去ブロック毎に、少なくとも１個以上有する。

【０１２７】この場合、均等ブロックあるいはブートブロック構成のフラッシュメモリのうちからシステムの目的に合わせた、あるいはより安価なフラッシュメモリを採用することが可能となる利点がある。

【０１２８】一実施形態では、再構築用に予め確保されている予備ブロックは、複数個の不揮発性半導体記憶装置に含まれるブロックサイズが異なる各消去ブロック毎に、少なくとも１個以上有する。

【０１２９】この場合、ファイルシステムが必要とする予備ブロックの数を少なくできる利点がある。このため、ファイルシステムが安価に構成できる利点もある。

【０１３０】一実施形態では、再構築用に予め確保されている予備ブロックは、消去ブロックのサイズ以上のサイズを有する。

【０１３１】この場合は、ブートブロック（例えば８Ｋバイトのサイズ）用の予備ブロックとして、８Ｋバイトより大きいサイズのブロック（例えば６４Ｋバイトのサイズ）を予備ブロックに割り当て、予備ブロックのサイズを一定にできるため、管理が簡単になる。

【０１３２】本発明のファイルシステムは、複数のセクタを有する不揮発性半導体記憶装置を用いたファイルシステムにおいて、該各セクタ毎に、セクタについての複数の状態情報のいずれかを格納する記憶手段と、該不揮発性半導体記憶装置のアクセス時に、該記憶手段内の該各セクタの状態情報に基いて、該不揮発性半導体記憶装置に既に格納されているデータを保証する保証手段とを具備し、前記セクタについての複数の状態情報として、「未使用」、「データ書き込み中」、「データ書き込み完了」、「データ有効」、「データ無効」という少なくとも５つの状態を設定しており、前記保証手段は、データが書き込まれるセクタの状態情報を「未使用」から「データ書き込み中」へ更新する手段と、該セクタへのデータの書き込みが完了した後、該セクタの状態情報を「データ書き込み中」から「データ書き込み完了」へ更新する手段と、該セクタにデータを新規に書き込んだ場合は、該セクタの状態情報を「データ書き込み完了」から「データ有効」へ更新し、該セクタに既存のデータの更新データを書き込んだ場合は、該既存のデータが書き込まれている他のセクタの状態情報を「データ有効」から「データ無効」に更新後、該更新データが書き込まれている該セクタの状態情報を「データ書き込み完了」から「データ有効」へ更新する。

【０１３３】本発明によれば、セクタについての５つの状態情報を記憶手段に格納している。このため、ファイルシステムが稼働中において、意図しない電力供給の遮断が生じても、電力供給が回復したときには、起動時の初期化処理に際し、セクタについての複数の状態情報に基いて、セクタの状態を正確に把握してきめ細かにフラッシュメモリを管理することができ、フラッシュメモリに格納されている既存のデータを破壊することなく、また既存のデータに矛盾を生じさせることなく、電力供給の遮断直前に実行中であった前段階へファイルシステムを自動的に回復させることが可能となる。

【０１３４】一実施形態では、前記消去ブロックについての複数の状態情報として、「未使用」、「データ有」、「ブロックフル」、「データ転送中」、「元ブロック消去中」、「元ブロック消去完了」という少なくとも６つの状態を設定しており、前記保証手段は、再構築用に予め確保されている予備消去ブロックの状態情報を「未使用」から「データ転送中」へ更新する手段と、データ転送元の消去ブロック内のデータを該予備消去ブロックへ転送して記憶させる手段と、該データの転送並びに記憶の完了後、該予備消去ブロックのブロック状態情報を「データ転送中」から「元ブロック消去中」へ更新し、該データ転送元の消去ブロックのデータを消去し、該予備消去ブロックのブロック状態情報を「元ブロック消去中」から「元ブロック消去完了」へ更新する手段と、データ転送元の消去ブロックを新たな予備消去ブロックとして確保し、この新たな予備消去ブロックの状態情報を「データ有」あるいは「ブロックフル」から「未使用」へと更新する手段と、データ転送先の該予備消去ブロックの状態情報を「元ブロック消去完了」から「データ有」へ変更する手段とを含む。

【０１３５】この様に、消去ブロックの状態情報として「未使用」、「データ有」、「ブロックフル」、「データ転送中」、「元ブロック消去中」、「元ブロック消去完了」という６つの状態を利用した場合、前記５つの状態を利用した場合よりも更に消去ブロックの状態を正確に把握してきめ細かにフラッシュメモリを管理することができ、フラッシュメモリに格納されている既存のデータを破壊することなく、また、既存のデータに矛盾を生じさせることなく、電力供給の遮断直前に実行中であった前段階へファイルシステムを自動的に回復させることが可能となる。

【０１３６】一実施形態では、前記消去ブロックについての複数の状態情報として、「未使用」、「データ有」、「データ転送中」、「元ブロック消去中」、「元ブロックフォーマット中」という少なくとも５つの状態を設定しており、前記保証手段は、再構築に予め確保されている予備消去ブロックの状態情報を「未使用」から「データ転送中」へ更新する手段と、データ転送元の消去ブロック内のデータを該予備消去ブロックへ転送して

記憶させる手段と、該データの転送並びに記憶の完了後、該予備消去ブロックのブロック状態情報を「データ転送中」から「元ブロック消去中」へ更新し、該データ転送元の消去ブロックのデータを消去し、このデータ転送元の消去ブロックを新たな予備消去ブロックとして確保し、この新たな予備消去ブロックの状態情報を「データ有」から「未使用」へ更新する手段と、データ転送先の該予備消去ブロックの状態情報を「元ブロック消去中」から「元ブロックフォーマット中」へ更新し、データ転送元の新たな予備消去ブロックへ新たな予備消去ブロックの消去可能回数などを書き込む（フォーマットする）手段と、フォーマット完了後、データ転送先の該予備消去法ブロックの状態情報を「元ブロックフォーマット中」から「データ有」へ更新する手段とを含む。

【０１３７】この様に、消去ブロックの状態情報として「未使用」、「データ有」、「データ転送中」、「元ブロック消去中」、「元ブロックフォーマット中」という５つの状態を利用した場合、消去ブロックの状態を正確に把握してきめ細かにフラッシュメモリを管理することができ、フラッシュメモリに格納されている既存のデータを破壊することなく、また既存のデータに矛盾を生じさせることなく、電力供給の遮断直前に実行中であった前段階へファイルシステムを自動的に回復させることが可能となる。

【０１３８】この場合、消去ブロックの状態情報として「ブロックフル」という状態を設定する前記実施形態と異なり、ブロック状態「ブロックフル」を設定しない方法は、消去ブロックが不均等なサイズのセクタに分割されて使用される場合に特に有効である。なぜならば、不均等なサイズのデータを同一の消去ブロック内の連続したセクタに書き込む場合において、次に書き込むべきデータのサイズが不明である場合に、消去ブロックの状態情報が「ブロックフル」となるかどうかの判断が難しいためである。

【０１３９】一実施形態では、各消去ブロックのブロック消去回数を平均化するために、各消去ブロック内に該消去ブロックの消去可能回数を記憶する。この消去可能回数をファイルシステムの再構築時において利用することにより、各消去ブロックの消去可能回数のバラツキを一定の回数内に抑え、フラッシュメモリとしての総書き替え回数を向上させることが可能になる。

【０１４０】また、消去ブロックの消去可能回数は、０から１つずつ増加させ、消去可能回数の仕様値の最大値までインクリメントする方法（この場合は消去可能回数でなく消去回数を表す）と、これとは逆に消去可能回数の仕様値の最大値から１つずつ減少させ、０までデクリメントする方法が考えられるが、後者の方法においては、消去可能回数が０になった時点で該消去ブロックの寿命が来たと判定できるため、消去可能回数の判定アルゴリズムを簡単にできる。

【０１４１】一実施形態における消去ブロックのフォーマット手段では、フォーマットした日時を記憶することも可能であり、この場合、一定時間経過後に消去ブロック内のデータのリフレッシュ動作を行うことが可能になる。

【図面の簡単な説明】

【図１】本発明のファイルシステムの一実施形態を示すブロック図である。

【図２】図１のファイルシステムのフラッシュメモリ部に格納されているデータの構成図である。

【図３】図１のファイルシステムにおけるファイルシステムメモリ部に格納されている各消去ブロックに関するデータの構成図である。

【図４】図１のファイルシステムにおけるファイルシステムメモリ部に格納されている各セクタに関するデータの構成図である。

【図５】図１のファイルシステムにおけるファイルシステムメモリ部に格納されている各消去ブロック毎に、各セクタの数をそれぞれのセクタ状態に応じて整理したデータの構成図である。

【図６】図１のファイルシステムにおけるデータの読み出し処理に関するフローチャートである。

【図７Ａ】図１のファイルシステムにおけるデータを書き込む処理に関するフローチャートである。

【図７Ｂ】図１のファイルシステムにおけるデータを書き込む処理に関する図７Ａに続くフローチャートである。

【図７Ｃ】図１のファイルシステムにおけるデータを書き込む処理に関する図７Ｂに続くフローチャートである。

【図８Ａ】図１のファイルシステムの再構築処理に関するフローチャートである。

【図８Ｂ】図１のファイルシステムの再構築処理に関する図８Ａに続くフローチャートである。

【図９Ａ】図１のファイルシステムの初期化に関するフローチャートである。

【図９Ｂ】図１のファイルシステムの初期化に関する図９Ａに続くフローチャートである。

【図９Ｃ】図１のファイルシステムの初期化に関する図９Ｂに続くフローチャートである。

【図９Ｄ】図１のファイルシステムの初期化に関する図９Ｃに続くフローチャートである。

【図１０】従来の技術におけるフラッシュメモリに格納されたデータの更新処理に関するフローチャートである。

【図１１】従来の技術におけるファイルシステムの再構築処理に関するフローチャートである。

【図１２】図１のファイルシステムにおけるフラッシュメモリ部に格納されているデータの構成図の内、ブロック状態情報のブロック状態を表す他の実施形態を示す図

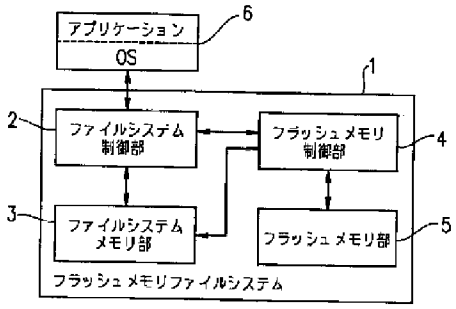
である。

【図１３】図１のファイルシステムの初期化に関する図９Ａ～図９Ｄの内、状態情報として「元ブロック消去完了」を設定した第１の実施形態に関わる部分のフローチャートを示す図である。

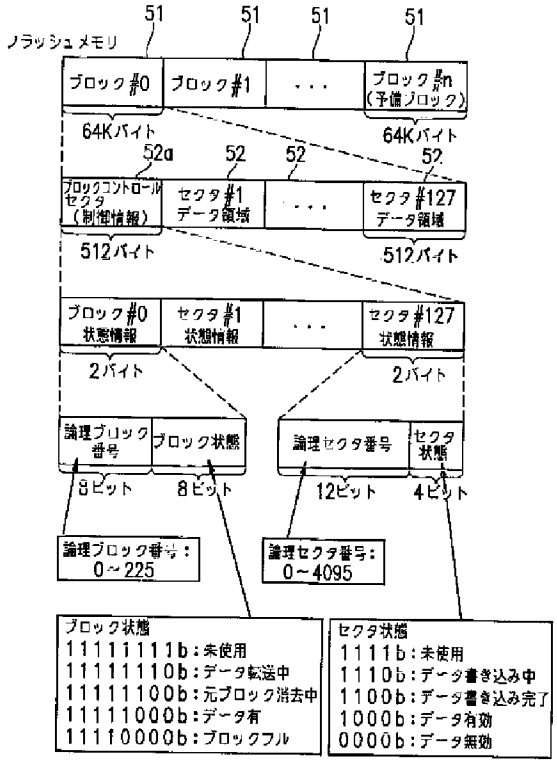
【符号の説明】

１　ファイルシステム
２　ファイルシステム制御部
３　ファイルシステムメモリ部
４　フラッシュメモリ制御部
５　フラッシュメモリ部
６　アプリケーション、ＯＳ（オペレーティングシステム）
５１　消去ブロック、予備消去ブロック
５２　セクタ
５２ａ　ブロックコントロールセクタ

【図１】



【図２】



【図３】

10

物理ブロック番号	論理ブロック番号	ブロック状態
0	0	データ有
1	1	データ・フル
⋮	⋮	⋮

【図４】

11

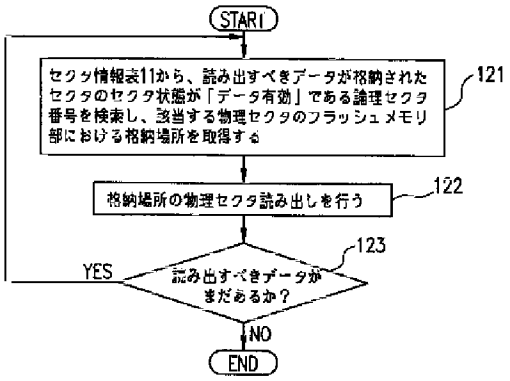
物理ブロック番号	物理セクタ番号	論理セクタ番号	セクタ状態
0	0	100	データ有効
0	1	10	データ無効
⋮	⋮	⋮	⋮

【図５】

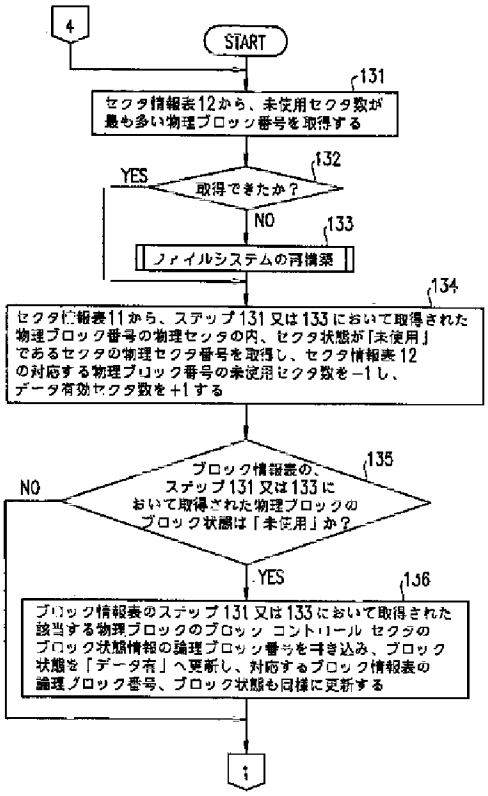
12

物理ブロック番号	未使用セクタ数	データ有効セクタ数	データ無効セクタ数
0	100	20	7
1	0	50	77
⋮	⋮	⋮	⋮

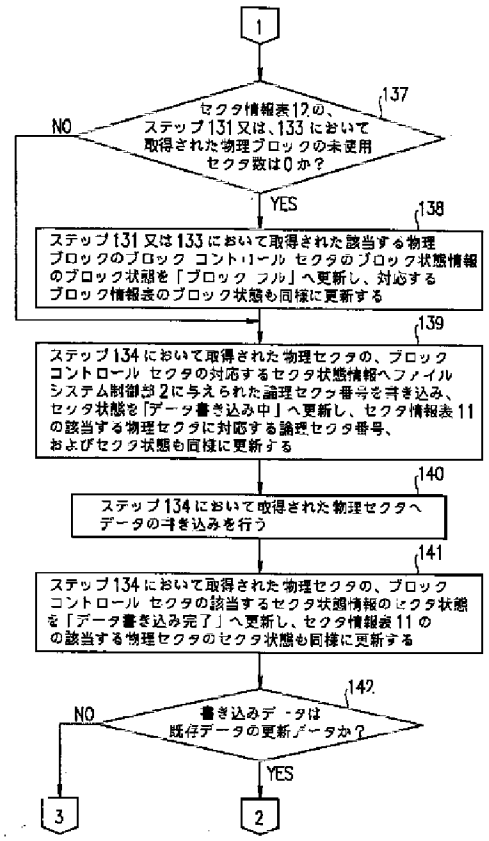
【図6】



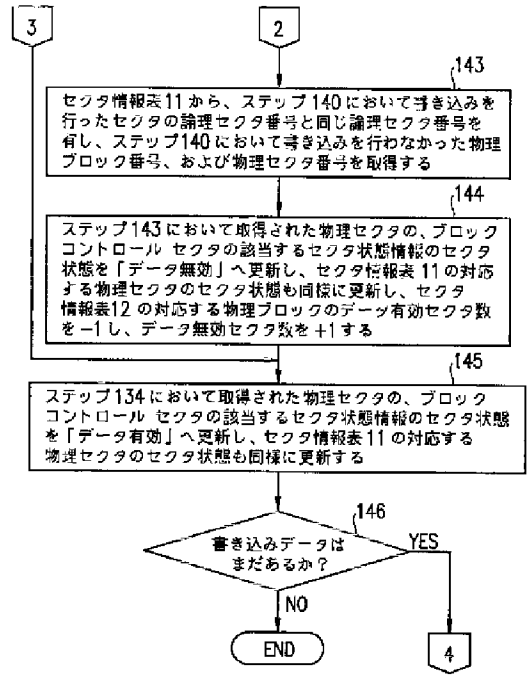
【図7A】



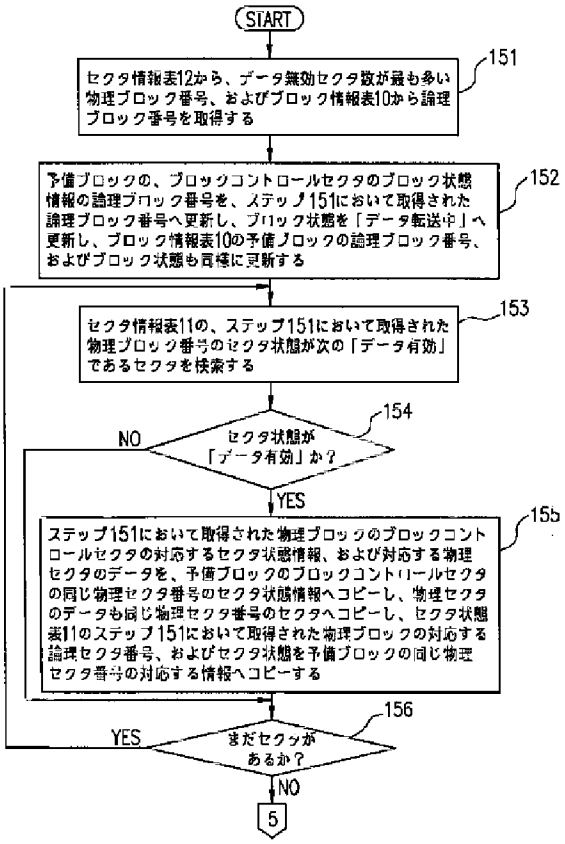
【図7B】



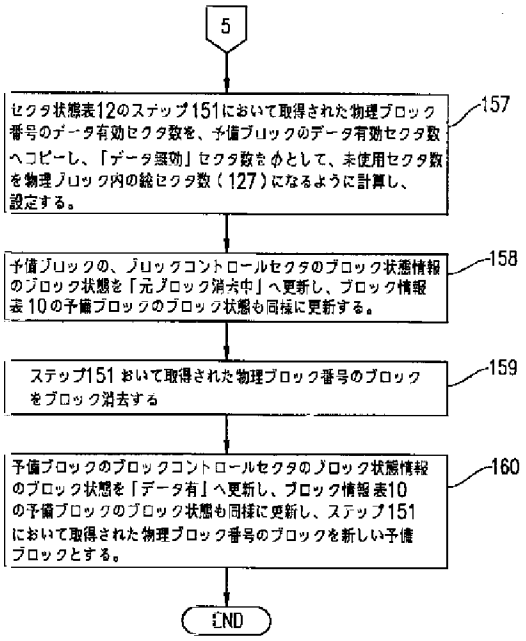
【図7C】



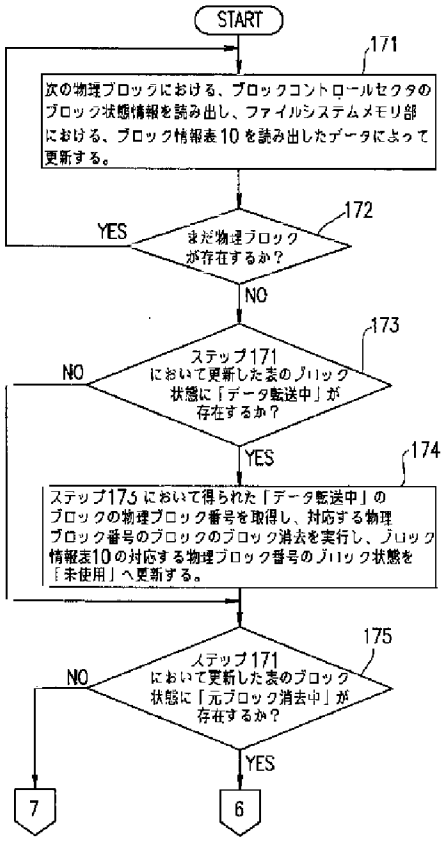
【図8A】



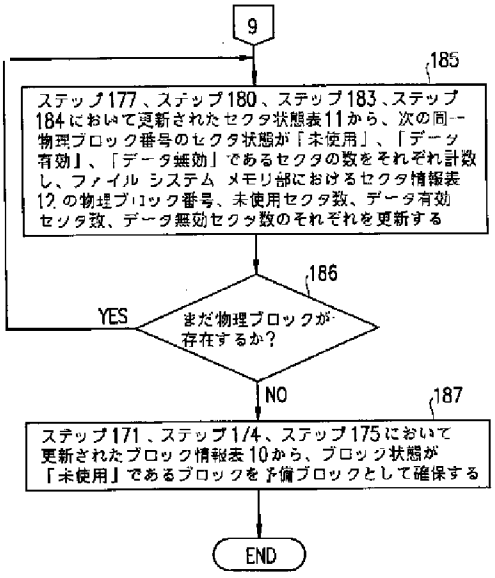
【図8B】



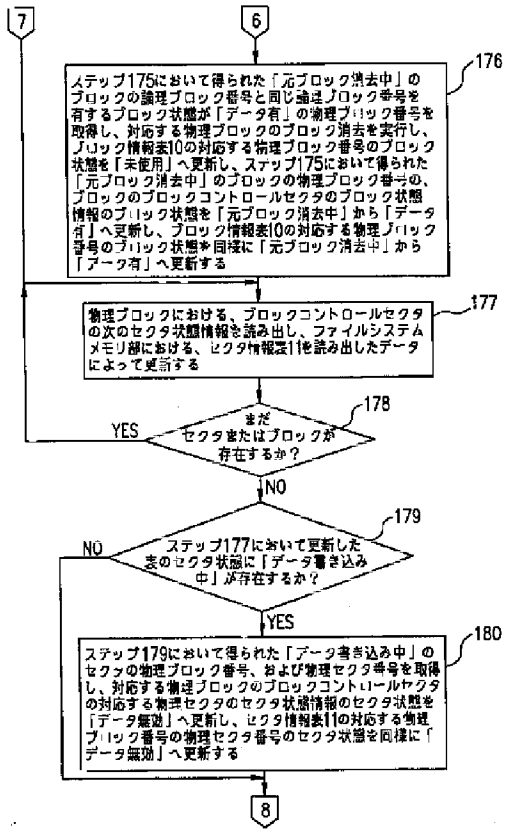
【図9A】



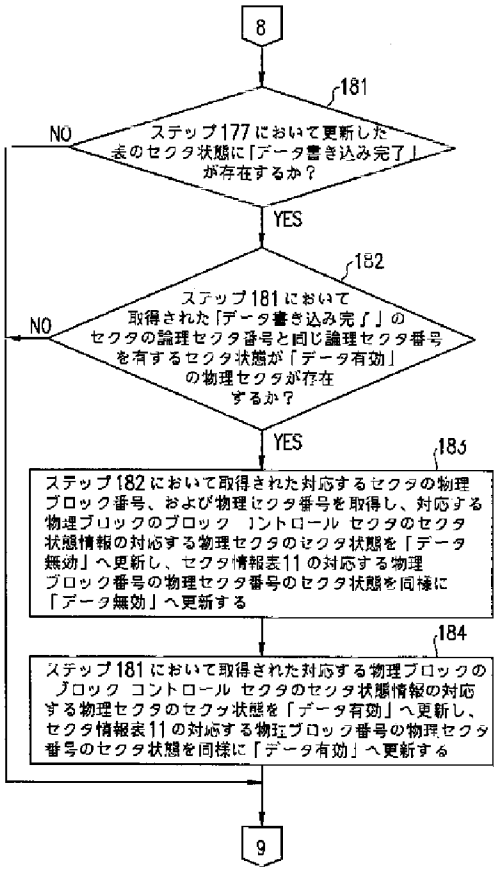
【図9D】



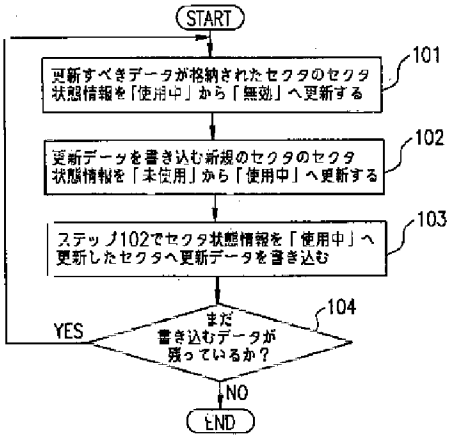
【図９Ｂ】



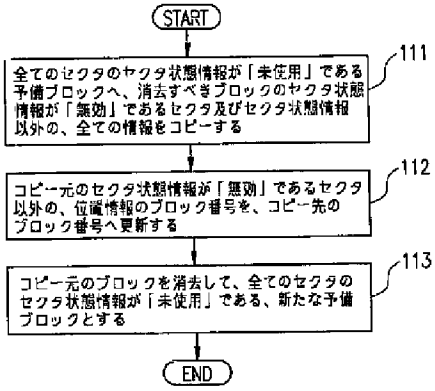
【図９Ｃ】



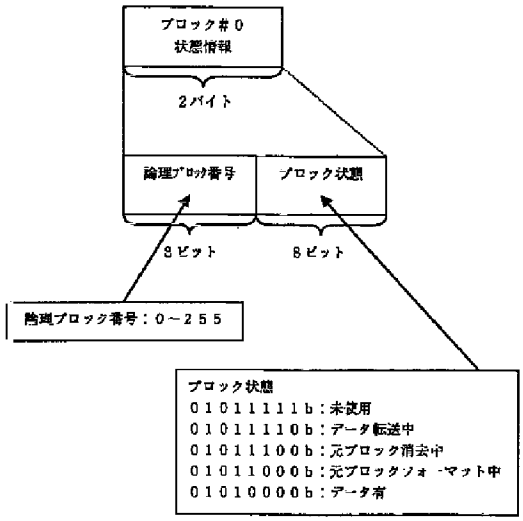
【図１０】



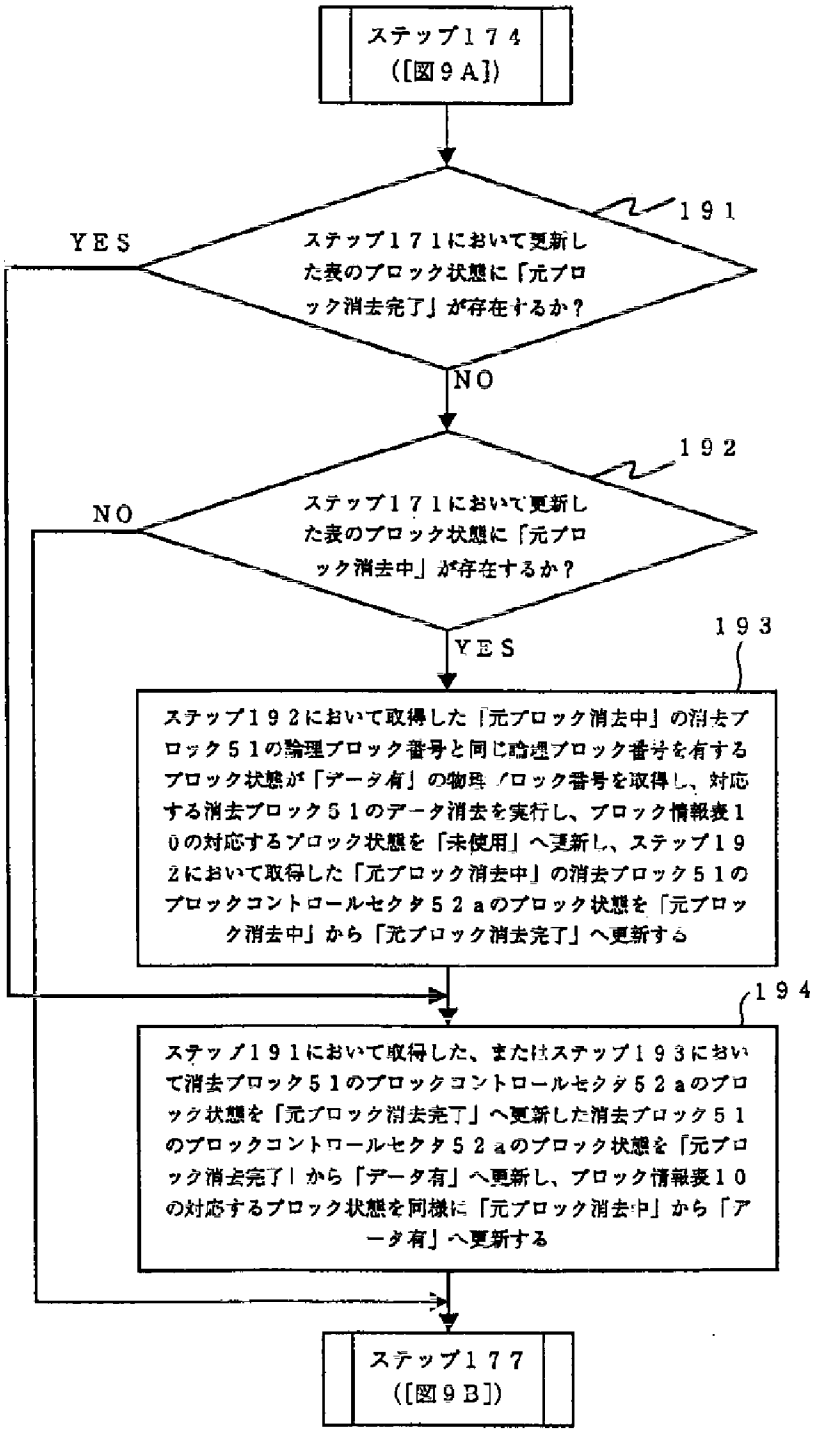
【図１１】



【図12】


ブロック#0
状態情報
2バイト
論理ブロック番号
ブロック状態
8ビット
8ビット
論理ブロック番号：0～255
ブロック状態
01011111b：未使用
01011110b：データ転送中
01011100b：元ブロック消去中
01011000b：元ブロックフォーマット中
01010000b：データ有

【図13】


ステップ174
([図9A])
191
ステップ171において更新した表のブロック状態に「元ブロック消去完了」が存在するか？
YES
NO
192
ステップ171において更新した表のブロック状態に「元ブロック消去中」が存在するか？
NO
YES
193
ステップ192において取得した「元ブロック消去中」の消去ブロック51の論理ブロック番号と同じ論理ブロック番号を有するブロック状態が「データ有」の物理ブロック番号を取得し、対応する消去ブロック51のデータ消去を実行し、ブロック情報表10の対応するブロック状態を「未使用」へ更新し、ステップ192において取得した「元ブロック消去中」の消去ブロック51のブロックコントロールセクタ52aのブロック状態を「元ブロック消去中」から「元ブロック消去完了」へ更新する
194
ステップ191において取得した、またはステップ193において消去ブロック51のブロックコントロールセクタ52aのブロック状態を「元ブロック消去完了」へ更新した消去ブロック51のブロックコントロールセクタ52aのブロック状態を「元ブロック消去完了」から「データ有」へ更新し、ブロック情報表10の対応するブロック状態を同様に「元ブロック消去中」から「データ有」へ更新する
ステップ177
([図9B])

Empty/full status tracking method for random access read/write memory cell, involves generating enabled signals to allow write and read flag status registers to be updated by using write and read flag enable logics, respectively

DOCUMENT ID
RD-527033-A

DATE PUBLISHED
2008-03-10

APPLICATION NO
2008RD-527033

DATE FILED
2008-03-10

FOREIGN APPLICATION PRIORITY DATA

COUNTRY	APPLICATION NO	APPLICATION DATE
RD	527033	

DERWENT ACCOUNT NO
RD-527033-A

DERWENT WEEK
200839

PATENT FAMILY

PUB NO	PUB DATE	LANGUAGE
RD 527033 A	2008-03-10	EN

ABSTRACTED PUB NO	CHOSEN DRAWING	DERWENT CLASS
RD 527033 A	Dwg.1/1	U14

TITLE TERMS
EMPTY FULL STATUS TRACK METHOD RANDOM ACCESS READ WRITING MEMORY CELL GENERATE ENABLE SIGNAL ALLOW FLAG REGISTER UPDATE LOGIC RESPECTIVE

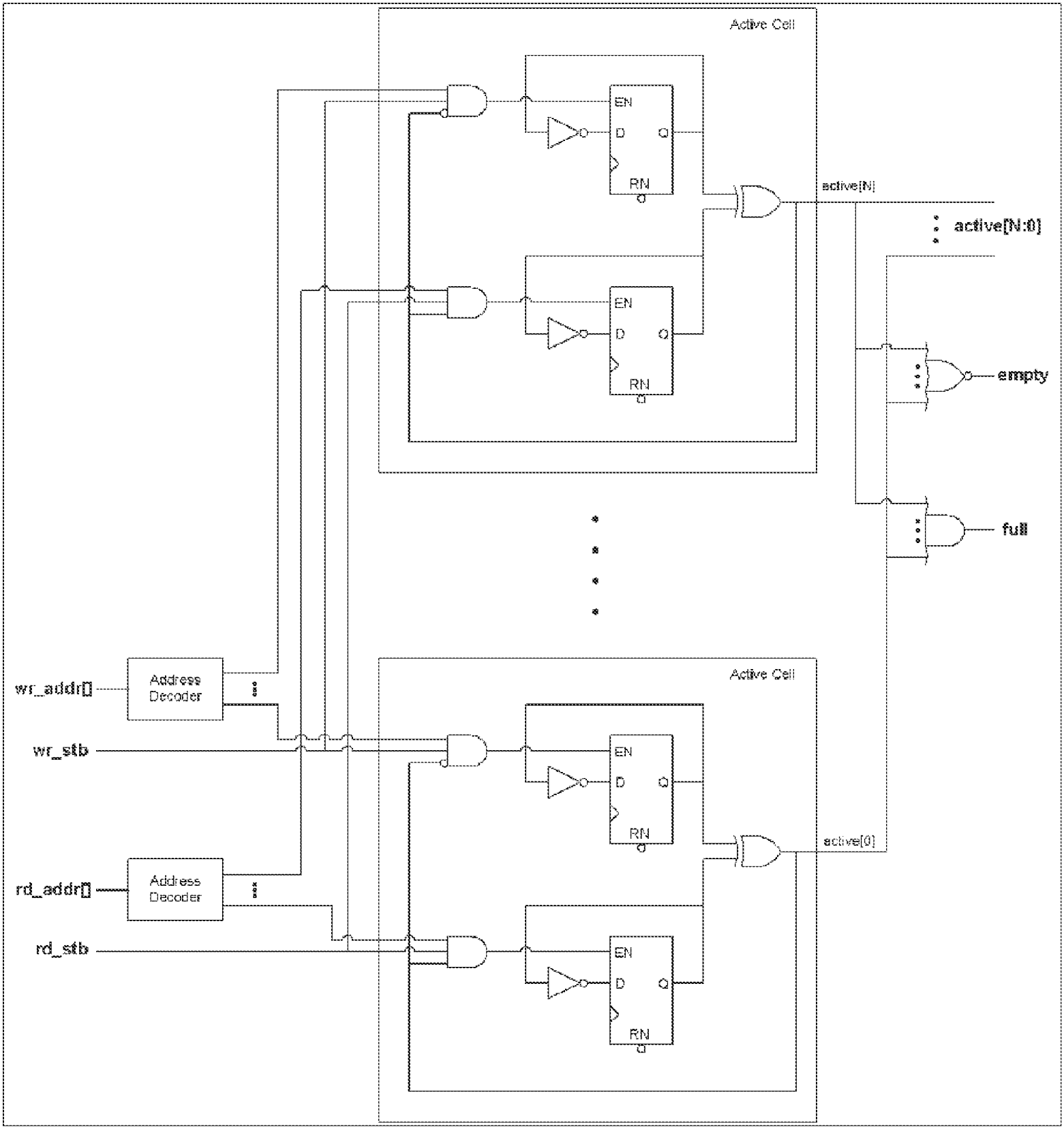
APPL PUB NO	APPL DESCRIPTOR	APPL NO	APPL DATE
RD 527033A	N/A	2008RD-527033	Sun Mar 09 20:00:00 EDT 2008

EPI CODES
U14-A03B4

Abstract

NOVELTY - The method involves selecting an appropriate active cell based on an input read address and a write address by using a read and write address decoder. Enabled signals are generated to allow write and read flag status registers to be updated by using write and read flag enable logics. The read or write flag status register is updated with an inverted version of a current state value of the read or write flag status register, when the read or write enable signal is asserted.
USE - Method for tracking empty/full status for a random access read/write memory cell by using an XOR gate in a verilog hardware description language.
ADVANTAGE - The method tracks the status of a memory cell in an efficient way without the need for arbitration, thus allowing simultaneous read and write operations to be taken place in a RAM without the need for arbiter logic to update the cell status.
DESCRIPTION OF DRAWING(S) - The drawing shows a circuit diagram of a random access read/write memory cell.

Copyright © 2022 Clarivate Analytics. All rights reserved. Republication or redistribution of Clarivate Analytics content, including by framing or similar means, is prohibited without the prior written consent of Clarivate Analytics. Clarivate and its logo are trademarks of the Clarivate Analytics group.


Active Cell
EN
D
Q
RN
active[N]
active[N:0]
empty
full
wr_addr[]
Address Decoder
wr_stb
rd_addr[]
Address Decoder
rd_stb
Active Cell
active[0]

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC ACKNOWLEDGEMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/931,506	**12/13/2022 05:03:03 PM ET**	**RMMS.003C12**

Title of Invention

STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	3894	FILED BY	Tayler Amerine
PATENT CENTER #	61292285	FILING DATE	09/12/2022
CUSTOMER #	20995	FIRST NAMED INVENTOR	Andrey V. Kuzmin
CORRESPONDENCE ADDRESS	-	AUTHORIZED BY	Aaron Davis

Documents

TOTAL DOCUMENTS: 14

DOCUMENT		PAGES	DESCRIPTION	SIZE (KB)
003C12_AMEND.pdf		12	-	82 KB
003C12_AMEND-A....pdf	(1-1)	1	Amendment/Request for Reconsideration-After Non-Final Rejection	15 KB
003C12_AMEND-CLM.pdf	(2-8)	7	Claims	38 KB
003C12_AMEND-SPEC.pdf	(9-9)	1	Specification	40 KB
003C12_AMEND-INTRVIEW.APP.pdf	(10-10)	1	Applicant summary of interview with examiner	14 KB
003C12_AMEND-REM.pdf	(11-12)	2	Applicant Arguments/Remarks	20 KB

			Made in an Amendment	
003C12_IDS.pdf		5	-	58 KB
003C12_IDS-TRAN.LET.pdf	(1-1)	1	Transmittal Letter	19 KB
003C12_IDS-IDS.pdf	(2-5)	4	Information Disclosure Statement (IDS) Form (SB08)	49 KB
FOR_JP2001051889A_WTRANS.PDF		39	Foreign Reference	1890 KB
FOR_JP2011123863A_WTRANS.PDF		37	Foreign Reference	542 KB
FOR_JP2011203916A_WTRANS.PDF		34	Foreign Reference	557 KB
FOR_KR20000025612A_WTRANS.PDF		13	Foreign Reference	1820 KB
FOR_RD527033.PDF		3	Foreign Reference	229 KB
NPL_Chang2010.PDF		10	Non Patent Literature	1472 KB
NPL_Seppanen2010.PDF		12	Non Patent Literature	160 KB

Digest

DOCUMENT	MESSAGE DIGEST(SHA-512)
003C12_AMEND.pdf	27DCDFCFC509AAB927D66F661F77F36F56F98711899A86B1964C344A27FE71B109F90182E0F8AAD6DA8C2F036628C96AFD2BFC20B971EB52DAA0E3846C91AEB4
003C12_AMEND-A...pdf	E8A9AC3DF0E2CD868355E5E9D5F2835F8DC216FCEA8E01BCFE91930659BFFB81C747BB6331DAA545FA5B91AB963096D433CDF49510875B0DD2CB1027DC32B364

003C12_AMEND-CLM.pdf	63895D98827AFE599CE8F93BF4DA3611458B002D0A407746EC DE63CD0AD001CBC0396515B0E397C8B620C73F94F44FC3A26 4276AE2509BC89BADC51BC36AD2A8
003C12_AMEND-SPEC.pdf	D52F0DC898EAE4559266EF42904B2578A235C19B0051C05AC BEE0D9370FE498303E6C67179F2D8B952E0D709B75B72FDF0 D06289E935EB2D41C0D8AB2837EA2D
003C12_AMEND-INTRVIEW.APP.pdf	D2D96192181DAB15FF349192F60735560835A8DD033BF4FCC D856A12F17CAD7BC0AECFE2AEE802E7412FFEDC6D7EE9209 87C464714824D03C2A873571A291BC2
003C12_AMEND-REM.pdf	29C19E24A598114FAED018044AB8C1C27E0664C67E83CEC90 BE29F31CDDA43FEE9278022A86068E7C9EE874FD211B19065 12E7144EE231E4384855FDCDD9B030
003C12_IDS.pdf	39432C9CB3F249ABDB9DB6F7B937B0FD76BF4CC5909245D77 38CEF73F372CE49691C3E28B073376988BCA17FA5F3357D748 EA18E2CBAE5601E63F0E8E6465753
003C12_IDS-TRAN.LET.pdf	669F5E8FF00185DB8AEAEF78AACF0B84341C82281199C1550 664B027CFA963D2D37E33D57BBAA1855A02D083A7735B2441 A58F60D9A0F63543266DC504EB7880
003C12_IDS-IDS.pdf	F4968C5A2D920FD478A8AB9172AC58A0C749C0CDE4CD6049 F737E1456CA76738EC900552C81E18C01089B8CD464DF5D97 56D50387CFAFECD7BE81B803386A757
FOR_JP2001051889A_WTRANS.PDF	52AF2B475A49D6253A7A26B7B4FFCEF96AC09FEB7D6F2F266 940215CD49BD989244A73320024A29AF9949503EDE297857AB C2476F11FC489FCA688D1090EE158
FOR_JP2011123863A_WTRANS.PDF	782B751106C36DB0DEE632F94B3F52F9C59624FB7EED0A5A8 78D07DE754BFB79666F64FB1966EDD71C813CFC74F815A05A 1938603DD3A5CE2FE86F0F19DFA949
FOR_JP2011203916A_WTRANS.PDF	6765513C4E7233A0EDE6245F9E3A4C6CE496543EDD270E790 57DE2D9DF3996E97457878BCDCCADC53839ED436BA7E306B 0029E43406FFB4A8BF058D075A9BCC3

FOR_KR20000025612A_WTRANS.PDF	D159056FDA015BD096554BFCB976F723C3CB4A058C841371C ECC50A4275094E916BB8FC77DF0C17D52A70028E8403171E0 6F263F2D0E186F8B4AA017C113F807
FOR_RD527033.PDF	5EBFBD96B39A42239893CADDCE200F50AF7D0DE33FC3AF25 96BC3F88A3835D941DAB7F2EF92C074DFC27E599C4BE9977C 5900BF5E3089AF45A41C7C618AE879F
NPL_Chang2010.PDF	8158C340CD428DA283DBD2B8AC62496EA2AF04BF1E96051A A74D3203258B32995D6C43D9EC6644F47EB149632FB9892E53 74CC05F4E09E7EEF2DABEAB8C08250
NPL_Seppanen2010.PDF	9E63CD864032EBA40AFCD47FB90DFEDE7DF9028AF87298686 2A16C14DC3421ECE1BC1271BD6226EB35073365D66F524594 787B7995FC18D7C8435B5AF179E63D

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 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.

JP2011123863A FLASH MEMORY STORAGE SYSTEM FOR SIMULATING REWRITABLE DISK DRIVE, FLASH MEMORY CONTROLLER, COMPUTER SYSTEM, AND METHOD THEREOF

Data originating from sources other than the EPO may not be accurate, complete, or up to date.

The wording below is an initial machine translation of the original publication. To generate a version using the latest translation technology, go to the original language text and use Patent Translate.

Flash memory storage system, flash memory controller, computer system and method thereof for simulating a rewritable disk device

[0001] The present invention generally relates to a flash memory storage system, and more particularly to a flash memory storage system for simulating a rewritable disk device, a flash memory controller, a computer system equipped with the flash memory storage system, and The present invention relates to a method for simulating a rewritable disc.

[0002] A rewritable digital versatile disc (DVD = Digital Versatile Disc) drive is an optical disc storage device. Due to its large capacity, DVDs are gradually being replaced by conventional CDs (Compact Discs) in order to store high-quality video data, high-quality audio data, and large amounts of data. DVDs must be accessed via an optical pickup in the DVD drive. However, due to the large volume and high power consumption of DVD drive 2002 (shown in FIG. 1A), small computers (eg, netbooks) typically cannot be equipped with any DVD drive, which is very inconvenient for the user. In addition, since the DVD drive 2002 uses a mechanical optical pickup to read data from or burn data to the DVD 2004, the access speed of the DVD drive 2002 is very slow. Furthermore, the volume of DVD 2004 is large and it is inconvenient to carry.

[0003] The above problem is that if the DVD drive 2002 is replaced with a flash drive 2006 having a flash memory, the flash memory data non-volatility, low power consumption, small volume and non-mechanical structure can be stored in the storage medium (storage medium).

[0004] A Universal Serial Bus (USB) storage device that uses a flash memory as its storage medium is disclosed in Patent Document 1, among which a USB storage device simulates a CD-ROM device and a USB storage device Data stored in can be prevented from being deleted by an accident. However, the user cannot satisfy the requirement of data rewritability because the data cannot be rewritten by using the CD-ROM device simulated by the USB storage device.

[0005] 米国特許第７，１１１，１２１号明細書

[0006] Accordingly, an object of the present invention is to provide a flash memory storage system capable of simulating a rewritable disk device.

[0007] An object of the present invention is to provide a flash memory controller capable of simulating a rewritable storage medium by using a flash memory chip.

[0008] An object of the present invention is to provide a computer system capable of accessing a flash memory storage device in accordance with a rewritable disk command.

[0009] An object of the present invention is to provide a method capable of simulating a rewritable disk by using a flash memory storage device.

[0010] Embodiments of the present invention provide a flash memory storage system including a flash memory chip, a connector, and a controller. A flash memory chip has a large number of substance blocks. The connector is arranged to be connected to the host system. A controller is coupled to the flash memory chip and the connector. The controller arranges a large number of logical blocks and maps the logical blocks to a part of the actual block (also called “mapping”). The controller further identifies at least one rewritable disk command

from the host system and writes data from the host system to the physical block mapped to at least a portion of the logical block according to the at least one rewritable disk command. .

[0011] The present invention provides a flash memory storage system including a flash memory chip, a connector, and a controller. A flash memory chip has a large number of substance blocks. The connector is arranged to be connected to the host system. A controller is coupled to the flash memory chip and the connector. The controller arranges a large number of logical blocks and maps the logical blocks to a part of the physical block. At least one management information of a rewritable disk device is stored in one of the physical blocks. By arranging at least one management information of the controller and the rewritable disk device, the rewritable disk device can be rewritten by the host when the flash memory storage system is connected to the host. It can be identified as a simple storage medium.

[0012] The present invention provides a flash memory controller for managing a large number of actual blocks of a flash memory chip. The flash memory controller includes a microprocessor unit, a flash memory interface unit, a host interface unit, and a memory management unit. A flash memory interface unit is coupled to the microprocessor unit and arranged to be coupled to the flash memory chip. A host interface unit is coupled to the microprocessor unit and arranged to be coupled to the host system. A memory management unit is connected to the microprocessor unit. A memory management unit arranges the logical block into a number of logical blocks and maps the logical block to a part of the physical block. In addition, the memory management unit identifies at least one rewritable disk command from the host system, and data from the host system is mapped to at least a part of the logical block according to the at least one rewritable disk command. To write to.

[0013] The present invention provides a computer system that includes a central processing unit (CPU), a main memory, and an embedded flash memory storage device. An embedded flash memory storage device is connected to the CPU and includes a flash memory chip and a controller, of which the flash memory chip has a number of substantial blocks and the controller is connected to the flash memory chip. A controller is arranged in a number of logical blocks and maps the logical blocks to a portion of the physical block. In addition, the controller identifies at least one rewritable disk command from the CPU, and writes data to a physical block mapped to at least a part of the logical block according to the rewritable disk command.

[0014] The present invention provides a computer system including a CPU, a main memory, a card reader, and a removable memory card. The card reader is connected to the CPU and has a card reader controller, and the card reader controller declares to the CPU that the card reader is a rewritable disk device. A removable memory card is plugged into the card reader. The removable memory card includes a flash memory chip and a memory card controller, wherein the flash memory chip has a number of substantial blocks, and the memory card controller is connected to the flash memory chip. A memory card controller is arranged in a number of logical blocks and maps the number of logical blocks to a portion of the physical block. In addition, the memory card controller identifies at least one rewritable disk command from the CPU, and writes data to a physical block mapped to at least a part of the logical block according to the rewritable disk command.

[0015] The present invention provides a flash memory storage system including a card reader and the removable memory card. The card reader includes a card reader controller, and the card reader controller is arranged to declare to the host system that the card reader is a rewritable disk device. A removable memory card is plugged into the card reader. The removable memory card includes a flash memory chip and a memory card controller, of which the flash memory chip has a number of substantial blocks and the memory card controller is connected to the flash memory chip. The memory card controller arranges a number of logical blocks and maps the logical blocks to a part of the physical block. The card reader controller identifies at least one rewritable disk command from the host system and writes data from the host system to a physical block mapped to at least a portion of the logical block according to the rewritable disk command.

[0016] The present invention provides a method of simulating a rewritable disk device by using a flash memory storage device. A flash memory storage device has a flash memory chip, of which the flash memory chip has a number of substantial blocks. In this method, a large number of logical blocks are arranged, and the logical block is mapped to a part of the substantial block, of which the logical block has a large number of logical addresses and the substantial block has a large number of substantial addresses And mapping the existing relationship between the logical address and the physical address. The present invention also includes accessing the flash memory storage device in response to the host system according to at least one management information of the rewritable disk device. The invention further includes calculating a write logical address and amount of data using the conversion parameters according to one write command of the at least one rewritable disk command provided by the host system, of which the write logic The address is one of logical addresses.

[0017] As described above, the flash memory storage device, the flash memory controller, the computer system and the method according to the embodiment of the present invention can simulate a rewritable disk by using the flash memory chip.

[0018] That is, in the embodiment of the present invention, the rewritable DVD device is simulated by using a flash memory storage device, and various data operations are simulated according to various rewritable DVD commands. Implemented on DVD device. Since the access speed of the flash memory storage system is faster than that of a general rewritable DVD device, the rewritable DVD device simulated by using the flash memory storage device is a general rewritable DVD device. Faster access speed. Also, a rewritable DVD device simulated by using a flash memory storage device does not require any mechanical optical pickup. Therefore, the size of the rewritable DVD device is greatly reduced. Thereby, a rewritable DVD device simulated by using a flash memory storage device can be placed in a small netbook.

[0019] It is a figure which shows the conventional rewritable disk apparatus and flash drive. 1 is a block diagram showing a host system and a flash memory storage device according to a first embodiment of the present invention. 1 is an explanatory diagram showing a computer, an input / output device, and a flash memory storage device according to an embodiment of the present invention. FIG. 1 is a block diagram illustrating a flash memory chip according to an embodiment of the present invention. It is explanatory drawing which shows the structure of the logic block concerning embodiment of this invention. 4 is a flowchart (flow chart) showing a method of simulating a rewritable digital versatile disc (DVD) according to an embodiment of the present invention. It is a block diagram which shows the flash memory storage system concerning 2nd Embodiment of this invention.

[0020] Hereinafter, embodiments for carrying out the present invention will be described with reference to the drawings. Generally speaking, a flash memory storage device (also called a flash memory storage system) includes a flash memory chip and a controller (also called a control circuit). The flash memory storage device is used together with the host system so that the host system can write data to and read data from the flash memory storage device. In addition, the flash memory storage device may also include embedded flash memory and software that is substantially executed on the host system and serves as a controller for the embedded flash memory.

[0021] FIG. 1B is a block diagram showing a host system and a flash memory storage device according to the first embodiment of the present invention.

[0022] In FIG. 1B, a host system 1000 includes a computer 1100 and an input / output (I / O) device 1106.

[0023] A computer 1100 includes a central processing unit (CPU) 1102, a main memory 1104, a system bus 1108, a data transmission interface 1110, and a main disk drive 1112. The computer 1100 can be a desktop computer or a notebook computer.

[0024] CPU 1102 interprets the mand and performs the main operations. Main memory 1104 is arranged for temporarily storing data and programs. A system bus 1108 is arranged for connection to various components of the host system 1000. A data transmission interface 1110 is arranged for connection to various peripheral devices. A main disk drive 1112 is arranged for installing an operation system (OS) 1122 and a burning program 1124. An OS 1122 is arranged to manage the host system 1000 hardware and software. In this embodiment, the OS 1122 may be Microsoft Windows® 7. However, the present invention is not limited to this. The burn program 1124 is arranged to provide a user interface that allows a user to burn data to the flash memory storage device 100 or erase data from the same.

[0025] As shown in FIG. 1C, the I / O device 1106 includes a mouse 1202, a keyboard 1204, a display 1206, and a printer 1208. It should be noted that the device shown in FIG. 1C is not intended to limit the I / O device 1106, and that the I / O device 1106 may further include other devices.

[0026] In this embodiment, the flash memory storage device 100 is coupled to other components of the host system 1000 via a data transmission interface 1110. The host system 1000 can write data to or read data from

the flash memory storage device 100 through operations of the CPU 1102, the main memory 1104, the I / O device 1106, and the main disk drive 1112. . The flash memory storage device 100 can be a flash drive 1212, a memory card 1214, a solid state driver (SSD) 1216 or a plug-in storage device 1218, as shown in FIG. 1C. It should be noted that if the flash memory storage device 100 is implemented as a pluggable storage device (ie, a pluggable storage device 1218), the pluggable flash memory storage device 100 may be the motherboard (not shown) of the host system 1000. N) directly.

[0027] The flash memory storage device 100 includes a connector 102, a flash memory controller 104, and a flash memory chip 106.

[0028] A connector 102 is coupled to the flash memory controller 104 and arranged to be coupled to the host system 1000. In this embodiment, the connector 102 is a universal serial bus (USB) connector. However, the present invention is not limited thereto, and the connector 102 may also be a parallel advanced technology attachment (Parallel Advanced Technology Attachment = PATA) connector, a serial advanced technology attachment (Serial Advanced Technology Attachment = SATA) connector, IEEE (Institute of Electrical and Electronics Engineers) 1394 connector, Peripheral Component Interconnect (PCI) high-speed connector, Secure Digital interface connector, Memory Stick (MS) interface connector, Multimedia card (Multi Media Card = MMC) interface connector, Compact Flash (registered trademark) (Compact Flash = CF) interface connector, Integrated Device Electronics (Integrated Device) Electronic = IDE) connector or other suitable connector.

[0029] The flash memory controller 104 is arranged in order to execute a number of logic gates or control commands implemented in the form of hardware or firmware, and variously in the flash memory chip 106 according to the commands of the host system 1000. Data manipulation. The flash memory controller 104 includes a microprocessor unit 202, a host interface unit 206 and a flash memory interface unit 208.

[0030] The microprocessor unit 202 is a main control unit of the flash memory controller 104, and it cooperates with the memory management unit 204, the host interface unit 206, and the flash memory interface unit 208 to perform various operations of the flash memory storage device 100. Execute.

[0031] A memory management unit 204 is coupled to the microprocessor unit 202 and arranged to perform a data write mechanism and a block management mechanism according to this embodiment. In particular, the memory management unit 204 declares to the host system 1000 that the flash memory storage device 100 is a rewritable digital versatile disc (DVD) device, and establishes management information for the rewritable DVD device. At the same time, it responds to the host system 1000 to access the flash memory storage device 100 based on the management information.

[0032] For example, the memory management unit 204 may notify the host system 1000 that the flash memory storage device 100 is a disk device that conforms to the DVD + RW standard, DVD-RW standard, DVD-RAM standard, or Blu-Ray Disc (BD) standard. Declare. Thereafter, the memory management unit 204 identifies the rewritable DVD command received from the host system 1000, and operates the flash memory chip 106 according to the rewritable DVD command. For example, the flash memory chip 106 burns data received from the host system 1000 to the flash memory chip 106 in accordance with a rewritable DVD command. That is, when the user of the host system 1000 operates the flash memory storage device 100 via a rewritable DVD device burning program, the memory management unit 204 identifies the burning command and performs the operation on the flash memory chip 106 according to the burning command. Perform various data operations. Hereinafter, the operation of the memory management unit 204 will be described in detail with reference to the accompanying drawings.

[0033] In this embodiment, the memory management unit 204 is implemented in the flash memory controller 104 in firmware form. For example, the memory management unit 204 includes a number of control commands burned into a program memory (for example, Read Only Memory = ROM), and the program memory is fitted into the memory management unit 204. When the flash memory storage device 100 is in operation, the control commands of the memory management unit 204 are executed by the microprocessor unit 202, and according to an embodiment of the present invention, a burn mechanism (eg, data read, write, erase) and flash Achieve a memory management mechanism.

[0034] In another embodiment of the present invention, the control command of the memory management unit 204 is stored as a program code in a specific area (for example, a system area in a flash memory chip for storing system data in Xiamen). Similarly, the control commands of the memory management unit 204 are executed by the microprocessor unit 202 when the flash memory storage device 100 is in operation. In yet another

embodiment, the memory management unit 204 is also implemented in the flash memory controller 104 as firmware.

[0035] A host interface unit 206 is coupled to the microprocessor unit 202 and receives and identifies commands and data from the host system 1000. That is, commands and data received from the host system 1000 are transmitted to the microprocessor unit 202 via the host interface unit 206. In this embodiment, the host interface unit 206 is a USB interface corresponding to the connector 102. However, the present invention is not limited thereto, and the host interface unit 206 may be a PATA interface, SATA interface, IEEE 1394 interface, PCI high-speed interface, SD interface, MS interface, MMC interface, CF interface, IED interface or There can be other suitable transmission interfaces.

[0036] A flash memory interface unit 208 is coupled to the microprocessor unit 202 and arranged to access the flash memory chip 106. That is, data written to the flash memory chip 106 is converted into a format that can be accepted by the flash memory chip 106 by the flash memory interface unit 208.

[0037] In the embodiment of the present invention, the flash memory controller 104 further includes a buffer memory 252, a power management unit 254, and an error correction unit 256.

[0038] A buffer memory 252 is coupled to the microprocessor unit 202 and temporarily stores data and commands received from the host system 1000 or data received from the flash memory chip 106.

[0039] A power management unit 254 is coupled to the microprocessor unit 202 and is arranged to control the power supply of the flash memory storage device 100.

[0040] An error correction unit 256 is coupled to the microprocessor unit 202 and arranged to perform an error correction procedure to ensure data accuracy. Specifically, when the memory management unit 204 receives a write command from the host system 1000, the error correction unit 256 generates an error checking and correcting (ECC) code for data corresponding to the write command. At the same time, the memory management unit 204 writes both the data and the corresponding ECC code to the flash memory chip 106. Subsequently, when the memory management unit 204 reads data from the flash memory chip 106, it also reads the ECC code corresponding to the data, and the error correction unit 256 performs an error correction procedure according to the ECC code.

[0041] A flash memory chip 106 is coupled to the flash memory controller 104 and arranged to store data. The flash memory chip 106 has a large number of substantial blocks. In the flash memory, each actual block is a minimum erase unit. That is, each actual block includes a minimum number of memory cells that are erased together. Each entity block has a number of entity addresses. In this embodiment, the entity address is an entity page. However, the present invention is not limited to this. Each entity page is a minimum program unit. That is, each physical page is a minimum unit for writing or reading data. Each physical page typically includes a user data area and a redundant area. The user data area stores user data, while the redundant area stores system data (eg, ECC code). In this embodiment, the flash memory chip 106 is a multi-level cell (Multi Level Cell = MLC) NAND flash memory chip. However, the present invention is not limited thereto, and the flash memory chip 106 may be a single level cell (Single Level Cell = SLC) NAND flash memory chip.

[0042] FIG. 2 is a block diagram showing a flash memory chip according to the embodiment of the present invention.

[0043] In FIG. 2, the flash memory chip 106 has actual blocks PBA (0) -PBA (N). Each entity block is a minimum erase unit. That is, each actual block includes a minimum number of memory cells that are erased together. Each entity block has 128 entity addresses (ie, entity pages). However, the invention is not so limited, and in other embodiments of the invention, each entity block may have 64, 256 or other suitable quantity of entity addresses.

[0044] The memory management unit 204 logically classifies the actual blocks PBA (0) -PBA (N) into a system area 302, a storage area 304, and a replacement area 306.

[0045] The logical blocks PBA (0) -PBA (S) logically belonging to the system area 302 are used to record system data such as the flash memory manufacturer and model number, and the physical address quantity corresponding to each physical block. used.

[0046] The entity blocks PBA (S + 1) -PBA (A) logically belonging to the storage area 304 (referenced in the center of FIG. 2) are used for storing data written by the host system 1000. That is, the flash memory storage device 100 actually stores the data received from the host system 1000 using the actual block in the storage area 304. Specifically, the entity block PBA (S + 1) -PBA (A) in the storage area 304 is further classified into a data area 304a and a spare area 304b, of which the entity block PBA (S + 1) -PBA (D) is an entity block that includes data, and an entity block PBA (D + 1) -PBA (A) in the spare area 304b is an entity block that has not yet been used for data storage. More specifically, when the memory management unit 204 obtains a real block from the spare area 304b and writes the data to the obtained real block, the obtained real block is associated with the data area 304a and associated with the data area 304a. When the actual block is erased, the erased actual block is associated with the spare area 304b.

[0047] The entity blocks PBA (A + 1) -PBA (N) logically belonging to the replacement area 306 are replacement entity blocks. For example, when the flash memory chip 106 is manufactured, 4% of the actual blocks are reserved for the purpose of replacement. That is, when the entity block in the system area 302 and the storage area 304 is damaged, the entity block reserved in the replacement area 306 can be used to replace the damaged entity block (ie, bad block). Thus, if there is still a normal entity block in the replacement area 306 and the entity block is damaged, the memory management unit 204 obtains a normal entity block from the replacement area 306 and replace. If there is no normal entity block in the replacement area 306 and the entity block is damaged, the flash memory storage device 100 is declared as a write protected state and cannot be used as write data. .

[0048] It should be noted that during the operation of the flash memory storage device 100, the classification relationship that the entity blocks PBA (0) -PBA (N) are classified into the system area 302, the data area 304a, the spare area 304b, and the replacement area 306, It is changing dynamically. That is, when the memory management unit 204 writes data to one entity block (for example, the entity block PBA (D + 2)) originally belonging to the spare area 304b, the entity block is associated with the data area 304a. Alternatively, since a substantial block in the data area 304a (or spare area 304b) is damaged and replaced with a substantial block obtained from the replacement area 306, the original block belonging to the replacement area 306 is replaced with the data area 304a (or spare area). Associated with region 304b).

[0049] The memory management unit 204 provides logical blocks LBA (0) -LBA (L) because the flash memory storage device 100 stores the data received from the host system 1000 by using the physical block in the storage area 304 instead of using it. To be accessed by the host system 1000. Specifically, each logical block includes a large number of logical addresses, and each logical address is mapped to an actual address of the actual block. Accordingly, the OS 1122 of the host system 1000 simply accesses the data according to the logical address, while the memory management unit 204 actually accesses the data in the corresponding physical address. In this embodiment, the memory management unit 204 maps the logical blocks LBA (0) -LBA (L) to the physical blocks in the storage area 304, and also creates a logical block-physical block mapping table. Record the map relationship between logical blocks and entity blocks. In this embodiment, the data written by the host system 1000 is written to the actual blocks belonging to the storage area 304, and the host system 1000 cannot access the actual blocks in the system area 302 and the replacement area 306.

[0050] FIG. 3 is an explanatory diagram showing the configuration of the logic block according to the embodiment of the present invention.

[0051] In FIG. 3, the memory management unit 204 classifies the logical blocks LBA (0) -LBA (L) into a first logical unit 360 and a second logical unit 380, and the first logical unit 360 is logical block LBA (0)-. The second logical unit 380 includes LBA (G) and the logical block LBA (G + 1) -LBA (L).

[0052] In this embodiment, the memory management unit 204 conforms to a DVD disc (for example, a DVD + RW standard, a DVD-RW standard, a DVD-RAM standard, or a Blue-Ray Disc (BD) standard) in which the first logical unit 360 can be rewritten. The second logical unit 380 is arranged as a mass storage device (for example, a mass storage device conforming to the File Allocation Table 32 (FAT32) format or the NTFS format).

[0053] In particular, in this embodiment, the flash memory storage device 100 is connected to the host system 1000, and the CPU 1102 of the host system 1000 identifies that the flash memory storage device 100 is composed of a rewritable DVD device and a mass storage device. .

[0054] Specifically, when the CPU 1102 of the host system 1000 requests a device feature from the flash memory storage device 100, the memory management unit 204 determines that the flash memory storage device 100 is in the mass storage category and the first logical unit. Declare to host system 1000 that it has 360 and second logical unit 380. Thereafter, when the CPU 1102 requests the device characteristics of the first logical unit 360 from the flash memory storage device 100 (for example, the CPU 1102 inquires about the device characteristics of the first logic unit 360 via the 0 × 12 command), the memory The management unit 204 declares to the host system 1000 that the first logical unit 360 is a rewritable DVD device (for example, the memory management unit 204 responds to the host system 1000 with a 0 × 05 message), and the CPU 1102 When requesting device features of the second logical unit 380 from the flash memory storage device 100, the memory management unit 204 declares that the second logical unit 380 is a mass storage device (eg, the memory management unit 204 is a host system). Respond to 1000 with a 0x00 message). Therefore, when the host system 1000 tries to burn to a rewritable DVD device (that is, the first logical unit 360) of the flash memory storage device 100, the memory management unit 204 performs the host command according to the disk command received from the host system 1000. Data from the system 1000 is written to the physical block mapped to the logical blocks LBA (0) -LBA (G) of the first logical unit 360. In addition, when the host system 1000 tries to save in the mass storage device (that is, the second logical unit 380) of the flash memory storage device 100, the data from the host system 1000 is stored in accordance with the disk command received from the host system 1000. Write to a physical block mapped to logical blocks LBA (G + 1) -LBA (L) of two logical units 380.

[0055] It should be noted that in this embodiment, the first logical unit 360 having a DVD format and the second logical unit 380 having a disk format, even if the memory management unit 204 can rewrite the logical blocks LBA (0) -LBA (L). However, the present invention is not limited to this. For example, in another embodiment of the present invention, the second logic unit 380 can also be arranged as a Compact Disk-Read Only Memory (DC-ROM). Alternatively, in yet another embodiment of the present invention, the second logical unit 380 can also be another rewritable DVD device, and the memory management unit 204 can be a rewritable DVD device by the second logical unit 380. (For example, a disk device conforming to the DVD + RW standard, the DVD-RW standard, the DVD-RAM standard, or the BD standard) is declared to the host system 1000, and the second logical unit 380 according to the rewritable DVD command. Access the data inside. Furthermore, in yet another embodiment of the present invention, the memory management unit 204 also arranges all the logical blocks LBA (0) -LBA (L) as a rewritable DVD device, or the logical block LBA (0) -LBA (L) is classified into more logical units, and these logical blocks are arranged as a rewritable DVD device, a mass storage device, or a compact disk read-only memory device, respectively.

[0056] Here, the memory management unit 204 of the flash memory controller 104 executes disk commands from the host system 1000 according to the normal operation pattern, and performs various data operations on the second logical unit 380. Also, when the flash memory storage device 100 receives a rewritable DVD command given to the first logical unit 360, the memory management unit 204 uses the first logical unit 360 in response to an appropriate message. Simulate a DVD. Hereinafter, the operation mechanism of the memory management unit 204 will be described in detail.

[0057] In the embodiment of the present invention, the CPU 1102 of the host system 1000 transmits a storage medium feature inquiry command (eg, a rewritable DVD command 0 × 46 command) to the flash memory storage device 100 to store the first logical unit 360. When obtaining media characteristics, the memory management unit 204 sends a single layer rewritable DVD message to the host system 1000. Accordingly, the CPU 1102 identifies the storage medium of the first logical unit 360 as a single layer rewritable DVD. It should be noted that in other embodiments of the present invention, the memory management unit 204 may transmit the corresponding storage medium feature according to the capacity of the first logic unit 360. For example, if the capacity of the first logical unit 360 is less than or equal to 4.7 GB, the memory management unit 204 sends a single layer rewritable DVD message to the host system 1000. If the capacity of the first logical unit 360 is greater than or equal to 8.5 GB, the memory management unit 204 sends a double layer rewritable DVD message to the host system 1000.

[0058] In the embodiment of the present invention, the CPU 1102 of the host system 1000 sends a capacity inquiry command (for example, a rewritable DVD command 0 × 25 command) to the flash memory storage device 100 to obtain the capacity of the first logical unit 360. When doing so, the memory management unit 204 sends the capacity value of the first logical unit 360 back to the host system 1000. For example, if the total capacity is 15 GB, the memory management unit 204 declares to the host system 1000 that the capacity of the first logical unit 360 is 15 GB. It should be noted that in other embodiments of the present invention, the memory management unit 204 may respond to the host system 1000 according to the rewritable DVD design

specification when the total capacity is supported. That is. For example, if the total capacity is 15 GB, the memory management unit 204 declares to the host system 1000 that the capacity of the first logical unit 360 is 4.7 GB, and the memory management unit 204 Declare it as a single layer rewritable DVD. Further, if the total capacity is 15 GB, the memory management unit 204 declares to the host system 1000 that the capacity of the first logical unit 360 is 8.5 GB. Declares that it is a dual layer rewritable DVD.

[0059] In an embodiment of the present invention, the memory management unit 204 establishes a rewritable DVD blank flag. Here, the rewritable DVD blank flag is one piece of management information of the rewritable DVD device, and records the accumulation state of the first logical unit 360. For example, if the flash memory storage device 100 was just manufactured and no data was written, the rewritable DVD blank flag is labeled as "blank status". If data has already been written to the first logical unit 360 of the flash memory storage device 100, the rewritable DVD blank flag is labeled as "non-blank status". In this embodiment, "0" in the rewritable DVD blank flag represents "blank state", while "1" represents "non-blank state". Specifically, when data is re-burned to the same storage address of a rewritable DVD, the entire rewritable DVD must first be erased. Thus, the host system 1000 clears the first logical unit 360 using a clear command (for example, a rewritable DVD command 0 × A1 command). However, since data is erased in the unit of the substantial block in the flash memory chip 106, the substantial block is erased only when the data is written into the substantial block. When the CPU 1102 sends a clear command to the flash memory storage device 100, the memory management unit 204 displays a rewritable DVD blank flag as "blank state" and responds to the host system 1000 that the clear command has already been executed. . It should be noted that the memory management unit 204 does not actually erase the physical block mapped to the first logical unit 360, so the speed at which the flash memory storage device 100 executes the clear command is a general rewrite. Possible DVD devices are much faster than the speed at which they do the same thing.

[0060] In addition, when the CPU 1102 sends a format command (eg, a rewritable DVD command 0 × 04 command) or a section closure command (eg, a rewritable DVD command 0 × 5B command) to the flash memory storage device 100, memory management Unit 204 marks the rewritable DVD blank flag as "non-blank".

[0061] In particular, the memory management unit 204 is rewritable when the CPU 1102 sends a read command (eg, a rewritable DVD command 0 × 28 command) to the flash memory storage device 100 to read data from the first logical unit 360. It is determined whether the DVD blank flag is "blank". If the rewritable DVD blank flag indicates "blank state", the memory management unit 204 sends a null value directly to the host system 1000. If the rewritable DVD blank flag indicates "non-blank", the memory management unit 204 is mapped to the first logical unit 360 for the host system 1000 according to the logical address indicated by the read command. Read data from entity block.

[0062] For example, in this embodiment, the memory management unit 204 stores the rewritable DVD blank flag in the actual block in the system area 302 (upper right of FIG. 2). However, the invention is not so limited, and in other embodiments of the invention, the rewritable DVD blank flag may also be added to other non-volatile memory (eg, flash memory controller 104). Can also be stored in a non-volatile memory located in the memory.

[0063] In an embodiment of the invention, the memory management unit 204 establishes a write capacity parameter. Here, the write capacity parameter is one of the management information of the rewritable DVD device, and the write capacity of the first logical unit 360 is recorded. Specifically, in the rewritable DVD device, data is sequentially burned to the storage address of the rewritable DVD. Accordingly, the host system 1000 uses the next writable inquiry command in the first logical unit 360 (for example, the 0 × 52 command of the rewritable DVD command) to write the next writable address in the first logical unit 360. Inquire about. When the CPU 1102 sends a write command (for example, a rewritable DVD command 0 × 2A command) to the flash memory storage device 100 and burns the data to the first logical unit 360, the memory management unit 204 sets the logical address to be written. The data is written to the physical block mapped to the corresponding logical block, and the write capacity parameter is updated according to the write data capacity. Also, when the CPU 1102 sends a format command or a clear command to the flash memory storage device 100, the memory management unit 204 initializes the write capacity parameter (for example, the write capacity parameter is reset to zero). In particular, when the CPU 1102 sends the next writable address inquiry command to the flash memory storage device 100, the memory management unit 204 responds to the host system 1000 with the next writable address according to the write capacity parameter.

[0064] It should be noted that when a general rewritable DVD device executes a section closing command on a rewritable DVD, a section closing occurs at the storage address of the rewritable DVD, and subsequent data is not burned after the section is closed. It is not to be. Thus, in this embodiment, when the CPU 1102 sends a section close command to the flash memory storage device 100, the memory management unit 204 still follows the size of the closed section, even if the host system 1000 does not write any data. Update the write capacity parameter.

[0065] Similarly, in this embodiment, the memory management unit 204 stores the write capacity parameter in the actual block in the system area 302 (upper right of FIG. 2). However, the present invention is not so limited, and in other embodiments of the present invention, the write capacity parameter may also be additionally placed in other non-volatile memory (eg, flash memory controller 104). Stored in a non-volatile memory).

[0066] In another embodiment of the present invention, the CPU 1102 of the host system 1000 sends a read command or a write command to the flash memory storage device 100 to read data from the first logical unit 360 or to send data to the first logical unit 360. When burning, the memory management unit 204 uses the conversion parameters to calculate the logical address and data capacity indicated by the read or write command. Specifically, the logical access unit has 2048 bytes in the DVD system, while the logical access unit has 512 bytes in the flash memory system. Therefore, the conversion parameter is set to 4 (= 2048/512). For example, when the CPU 1102 tries to read 512 units of data starting from the logical address "512", the memory management unit 204 starts from the physical address mapped to the logical address "2048". Read data.

[0067] In the embodiment of the present invention, when the CPU 1102 of the host system 1000 sends an error correction data inquiry command to the flash memory storage device 100, the memory management unit 204 sends a non-support message to the host system 1000. Send back. Specifically, the DVD system uses an appropriate error correction algorithm to ensure the accuracy of the data stored on the DVD. In this embodiment, the DVD system is not suitable for a flash memory system. Thus, the error correction unit 256 (FIG. 1B) ensures data accuracy using an Error Checking and Correcting (ECC) code suitable for flash memory systems. Therefore, when the CPU 1102 attempts to read error correction data from the first logical unit 360, it rejects the error correction data inquiry command from the host system 1000.

[0068] In the DVD system, the host system 1000 uses a minute / second / frame (Minute / Second / Frame = MSF) read command (for example, the 0 × B9 command of the rewritable DVD command) to set the parameter of minute / second / frame. Can read data. In the embodiment of the present invention, when the CPU 1102 of the host system 1000 transmits the MSF read command to the flash memory storage device 100, the memory management unit 204 is indicated by the MSF read command through the following formula (1). Calculate logical address. $LBA = ((M \times 60 + S\text{-}2) \times 75 + F) \times 4$ (1)

[0069] Where LBA represents the logical address corresponding to the MSF read command in the flash memory, M represents the minute parameter of the MSF read command, S represents the second parameter of the MSF read command, and F represents the frame of the MSF read command. Represents a parameter. That is, one minute of data is stored in a 18000 (= 60 × 75 × 4) entity block, each entity address has 512 bytes, and one frame of data is stored in four entity addresses. In addition, each entity address has 512 bytes.

[0070] In the DVD system, the host system 1000 ejects a rewritable DVD in a rewritable DVD device using a disc eject command. In particular, after the rewritable DVD has been burned in the DVD system, the information stored in the DVD has changed, so the host system 1000 requests that the DVD be ejected, and the DVD device is rewritable again. After the insertion, the stored information on the DVD is read again. However, in the flash memory storage system, all stored information (eg, logical block-substance block map table) is temporarily stored in the buffer memory 252 (FIG. 1B). Thereby, the memory management unit 204 can obtain the latest information of the first logical unit 360 without having to unplug and re-plug the flash memory storage device 100. In an embodiment of the present invention, when the CPU 1102 of the host system 1000 sends a disk eject command to the flash memory storage device 100, the memory management unit 204 responds to the host system 1000 with a received message, but no action is taken. Absent. Subsequently, when the CPU 1102 queries the first logical unit 360, the memory management unit 204 returns a ready message directly to the host system 1000.

[0071] In an embodiment of the present invention, the memory management unit 204 sets a write protection flag to record whether the first logic unit 360 is in a "write protected state" or a "non-write protected state". Specifically, the user can set the non-write protection flag of the first logical unit 360 by using a vendor (vendor manufacturer) command through an application program. For example, if the user does not want to store data in

the first logical unit 360 and deletes it by mistake, the user can issue a vendor command via an application program installed in the host system 1000 to 204 can be requested to set the write protection flag as "write protected state".

[0072] In another embodiment of the invention, when changing the write protection flag, the application program asks the user for an authentication password and sends the authentication password and the data to be set to the memory management unit 204. The memory management unit 204 then compares the authentication password entered by the user with the vendor password. The memory management unit 204 changes the write protection flag according to the user's request only when the authentication password entered by the user matches the vendor password. Here, the memory management unit 204 stores the write protection flag and the vendor password in the system area 302 (upper right of FIG. 2). However, the invention is not so limited, and in other embodiments of the invention, the write protection flag and the vendor password may also be added to other non-volatile memory (eg, flash memory controller 104). Can also be stored in a non-volatile memory that is allocated to the system.

[0073] In the embodiment of the present invention, when the CPU 1102 of the host system 1000 sends a clear command, a format command or a section close command to the flash memory storage device 100, the memory management unit 204 indicates that the write protection flag is "write protected state". Determine if there is. When the write protection flag indicates "write protected state", the memory management unit 204 sends an error message back to the host system 1000 or does not take any action. The error messages are "write protected (ie ASC code 0 × 27 and SenseKey code 0 × 07)", "medium not present (ie ASC code 0 × 3A and SenseKey code 0 × 02)", " Medium is changing (ie ASC code 0 × 28 and SenseKey code 0 × 06) "," Command not supported (ie ASC code 0 × 20 and SenseKey code 0 × 05) "," Command parameter error (That is, ASC code 0 × 24 and SenseKey code 0 × 05) "," Data write error (ie, ASC code 0 × 03 and SenseKey code 0 × 03) "or" Data write error (ie, ASC code 0 × 11 and SenseKey code 0 × 03) ".

[0074] In addition to responding to the host system 1000 with one of the error messages described above, if the write protection flag indicates "write protection status", in another embodiment of the present invention, the memory management unit 204 also The storage system characteristics can be changed directly to prevent the host system 1000 from issuing any write or clear command. For example, when the CPU 1102 of the host system 1000 sends a storage medium feature inquiry command to the flash memory storage device 100, the memory management unit 204 determines whether the write protection flag indicates "write protected status". If the write protection flag indicates "write protected state", the memory management unit 204 returns a single layer read-only DVD message or a dual layer read-only DVD message to the host system 1000 for the first logical unit 360. Notifies the host system 1000 that is a read-only DVD.

[0075] FIG. 4 is a flowchart (flow chart) showing a method for simulating a rewritable digital versatile disc (DVD) according to an embodiment of the present invention.

[0076] In FIG. 4, during step S401, the memory management unit 204 arranges a large number of logical blocks (for example, LBA (0) -LBA (L)) and maps the logical blocks to a part of the actual blocks of the flash memory chip 106. (FIG. 2).

[0077] In step S403, the memory management unit 204 classifies a part of the logical block into one logical unit (for example, the first logical unit 360) (FIG. 3). In step S405, the memory management unit 204 arranges the logical unit as a rewritable DVD storage medium.

[0078] Next, in step S407, the memory management unit 204 declares to the host system 1000 that the logical unit is a rewritable DVD device. In step S409, the memory management unit 204 identifies that the rewritable DVD command has been received from the host system 1000, and the data received from the host system 1000 in accordance with the rewritable DVD command is mapped to a logical block. To record. Since the mechanism for identifying the rewritable DVD command by the memory management unit 204 has been described above, it will not be described here.

[0079] FIG. 5 is a block diagram showing a flash memory storage system according to the second embodiment of the present invention.

[0080] In FIG. 5, a flash memory storage system 500 includes a card reader 502 and a removable memory card 504.

[0081] The card reader 502 includes a connector 512, a card reader controller 514, and a memory card connection interface unit 516.

[0082] A connector 512 is arranged to connect to the host system 1000 (FIG. 1B). In this embodiment, the connector 512 is a USB connector. However, the present invention is not so limited, and the connector 512 can also be a PATA connector, SATA connector, IEEE 1394 connector, PCI high-speed connector, IED connector or other suitable transmission connector. In this embodiment, a card reader 502 is coupled to the host system 1000 in a detachable manner via a connector 512. However, the invention is not so limited, and in other embodiments of the invention, the card reader 502 can be coupled to the host system 1000 in a fixed manner.

[0083] A card reader controller 514 is coupled to the connector 512 and is arranged to control the operation of the card reader 502. In particular, in this embodiment, the card reader controller 514 declares to the host system 1000 that the card reader 502 is a rewritable DVD device. For example, the card reader controller 514 declares to the host system 1000 that the card reader 502 is a disk device that conforms to the DVD + RW standard, the DVD-RW standard, the DVD-RAM standard, or the BD standard. Further, the card reader controller 514 identifies the rewritable DVD command received from the host system 1000 and operates the removable memory card 504 according to the rewritable DVD command. The mechanism for responding to a rewritable DVD command received from the host system 1000 has been described above and will not be described here.

[0084] A memory card articulation interface unit 516 is coupled to the card reader controller 514 and arranged to couple to a removable memory card 504.

[0085] A removable memory card 504 is plugged into the card reader 502 in a separable manner. A removable memory card 50 includes a memory card controller 522 and a flash memory chip 524.

[0086] Since the memory card controller 522 arranges the logical blocks and maps the logical blocks to the physical blocks, the host system 1000 can access the flash memory chip 524 in the removable memory card 504 via the card reader 502.

[0087] Specifically, the card reader controller 514 declares to the host system 1000 that the removable memory card 504 is a rewritable DVD, and identifies a rewritable DVD command from the host system 1000, The memory card controller 522 executes the block management mechanism of the flash memory chip 524, and operates the flash memory chip 524 according to the command of the card reader controller 514.

[0088] In this embodiment, since the flash memory chip 524 is the same as the flash memory chip 106, the structure and function of the flash memory chip 524 are not described here.

[0089] As described above, the present invention has been disclosed by the embodiments. However, the present invention is not intended to limit the present invention, and is within the scope of the technical idea of the present invention so that those skilled in the art can easily understand. Therefore, the scope of patent protection should be defined based on the scope of claims and the equivalent area.

[0090] 100 Flash memory storage device 102 Connector 104 Flash memory controller 106 Flash memory chip 202 Microprocessor unit 204 Memory management unit 206 Host interface unit 208 Flash memory interface unit 252 Buffer memory 254 Power management unit 256 Error correction unit 302 System area 304 Storage area 304a Data area 304b Spare area 306 Replacement area 360 First logical unit 380 Second logical unit 502 Card reader 504 Removable memory card 512 Connector 514 Card reader controller 516 Memory card connection interface unit 522 Memory card controller 524 Flash memory chip 1000 Host system 1100 Computer 1102 Central processing unit 1104 Main memory 1106 Input / output device 1108 System bus 1110 Data transmission interface 1112 Main disk drive 1122 Operation system (OS) 1124 Burning program 1202 Mouse 1204 Keyboard 1206 Display 1208 Printer 1212 Flash drive 1214 Memory card 1216 Solid state driver (Solid State Driver = SSD) 1218 Insertable storage device 2002 Digital Versatile Disc (DVD) drive 2004 Digital Versatile Disc (DVD) 2006 Flash drive PBA (0) -PBA (N) Physical block LBA (0) -LBA (L) logical block

(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2011-123863
(P2011-123863A)
(43)公開日　平成23年6月23日(2011.6.23)

(51)Int.Cl.		F I		テーマコード（参考）
G06F 12/00	(2006.01)	G06F 12/00	514E	5B035
G06F 12/02	(2006.01)	G06F 12/02	570A	5B058
G06K 19/07	(2006.01)	G06F 12/00	597U	5B060
G06F 3/06	(2006.01)	G06K 19/00	N	5B065
G06F 3/08	(2006.01)	G06F 3/06	301Z	5B082

審査請求 有　請求項の数 38　OL　外国語出願　（全 26 頁）　最終頁に続く

(21)出願番号　特願2010-117729 (P2010-117729)
(22)出願日　平成22年5月21日 (2010.5.21)
(31)優先権主張番号　098142336
(32)優先日　平成21年12月10日 (2009.12.10)
(33)優先権主張国　台湾 (TW)

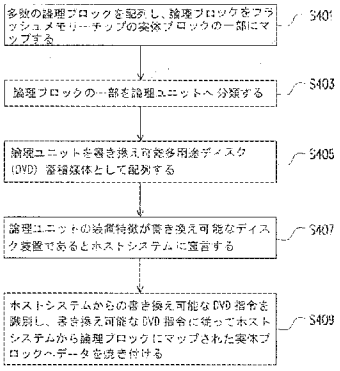
（特許庁注：以下のものは登録商標）
1．MEMORY　STICK

(71)出願人　509285089
群聯電子股▲ふん▼有限公司
台湾３５０苗栗縣竹南鎮群義路１號
(74)代理人　100147485
弁理士　杉村　憲司
(74)代理人　100143568
弁理士　英　貴
(74)代理人　100134577
弁理士　石川　雅章
(72)発明者　伍　漢維
台湾苗栗縣竹南鎮群義路１號
(72)発明者　黄　憲翔
台湾新竹市香山区頂埔路７８號

最終頁に続く

(54)【発明の名称】書き換え可能ディスク装置をシミュレーションするためのフラッシュメモリー記憶システム、フラッシュメモリー制御器、コンピューターシステムおよびその方法

(57)【要約】
【課題】書き換え可能ディスク装置をシミュレートするフラッシュメモリー記憶システムを提供する。
【解決手段】フラッシュメモリーチップとコネクターと制御器とを含むフラッシュメモリー記憶システムが提供される。フラッシュメモリーチップが多数の実体ブロックを有する。コネクターがホストシステムに連結されるよう配列される。制御器がフラッシュメモリーチップおよびコネクターに連結される。制御器が多数の論理ブロックを配列するとともに、論理ブロックを実体ブロックの一部にマップする。また、制御器が更にホストシステムからの少なくとも１つの書き換え可能ディスク指令を識別するとともに、書き換え可能ディスク指令に従ってホストシステムからのデータを論理ブロックの少なくとも１つにマップされたへ実体ブロックに書き込む。従って、書き換え可能ディスクがフラッシュメモリー記憶システムを使用することによりシミュレーションされる。
【選択図】図４



【特許請求の範囲】
【請求項１】
多数の実体ブロックを有するフラッシュメモリーチップと、
ホストシステムに連結されるよう配列されるコネクターと、
前記フラッシュメモリーチップおよび前記コネクターに連結され、かつ多数の論理ブロックを配列されるとともに、前記論理ブロックを前記実体ブロックの一部にマップする制御器とを備え、
そのうち、前記制御器が更に前記ホストシステムからの少なくとも１つの書き換え可能ディスク指令を識別するために配列されるとともに、少なくとも１つの前記書き換え可能ディスク指令に従って前記ホストシステムからのデータを前記論理ブロックの少なくとも一部にマップされたへ実体ブロックに書き込む、フラッシュメモリー記憶システム。
【請求項２】
前記制御器が、書き換え可能なディスク装置の少なくとも１つの管理情報を確立するとともに、前記書き換え可能なディスク装置の少なくとも１つの管理情報を前記実体ブロックの１つに保存する請求項１記載のフラッシュメモリー記憶システム。
【請求項３】
前記書き換え可能なディスク装置の前記した少なくとも１つの管理情報が、蓄積状態情報を含むとともに、前記蓄積状態情報が、前記論理ブロックの少なくとも一部の蓄積状態を記録する請求項２記載のフラッシュメモリー記憶システム。
【請求項４】
前記書き換え可能なディスク装置の前記した少なくとも１つの管理情報が、書き込み容量パラメーターを含むとともに、前記書き込み容量パラメーターが、前記論理ブロックの少なくとも一部の書き込み容量を記録する請求項２記載のフラッシュメモリー記憶システム。
【請求項５】
フラッシュメモリー記憶システムであり、
多数の実体ブロックを有するフラッシュメモリーチップと、
ホストシステムに連結されるよう配列されるコネクターと、
前記フラッシュメモリーチップおよび前記コネクターに連結され、かつ多数の論理ブロックを配列されるとともに、前記論理ブロックを前記実体ブロックの一部にマップする制御器とを備え、
そのうち、前記実体ブロックの１つが書き換え可能なディスク装置の少なくとも１つの管理情報を保存するとともに、前記フラッシュメモリー記憶システムが前記制御器および前記書き換え可能なディスク装置の少なくとも１つの管理情報を配列することにより前記ホストに連結される時、前記フラッシュメモリー記憶システムが、前記ホストが前記論理ブロックの少なくとも一部を前記書き換え可能なディスク装置の書き換え可能な蓄積媒体として識別することを許すものである、フラッシュメモリー記憶システム。
【請求項６】
前記制御器は、前記制御器が前記ホストシステムから蓄積媒体特性照会指令を受信した時、単一層書き換え可能DVDメッセージまたは二重層書き換え可能DVDメッセージを前記ホストシステムへ送り返すものである請求項５記載のフラッシュメモリー記憶システム。
【請求項７】
前記制御器は、前記ホストシステムから容量照会指令を受信した時、容量値を前記ホストシステムへ送り返すものであり、
そのうち、前記容量値が前記フラッシュメモリーチップより小さいか等しいものである請求項５記載のフラッシュメモリー記憶システム。
【請求項８】
さらに、不揮発メモリーを含み、そのうち、前記書き換え可能なディスク装置の少なくとも１つの管理情報が、前記不揮発メモリー中に保存されるものである請求項５記載のフラッシュメモリー記憶システム。

【請求項９】
前記書き換え可能なディスク装置の少なくとも１つの管理情報が、蓄積状態情報を含むとともに、前記蓄積状態情報が前記論理ブロックの少なくとも一部の蓄積状態を記録するものである請求項５記載のフラッシュメモリー記憶システム。
【請求項１０】
前記制御器が前記ホストシステムからクリアー指令を受信する時、前記制御器が前記蓄積状態情報をブランク状態としてマークするものである請求項９記載のフラッシュメモリー記憶システム。
【請求項１１】
前記制御器が前記ホストシステムからフォーマット指令またはセクション閉鎖指令を受信する時、前記蓄積状態情報を非ブランク状態としてマークするものである請求項９記載のフラッシュメモリー記憶システム。
【請求項１２】
前記制御器が前記ホストシステムから読み出し指令を受信する時、前記制御器は、前記蓄積状態情報がブランク状態を指示するか否か判定するものであり、
そのうち、前記蓄積状態情報が前記ブランク状態を指示する時、前記制御器が前記ホストシステムへヌル値を送り返すものである請求項９記載のフラッシュメモリー記憶システム。
【請求項１３】
前記書き換え可能なディスク装置の少なくとも１つの管理情報が、書き込み容量パラメーターを含むとともに、前記書き込み容量パラメーターが前記論理ブロックの少なくとも一部の書き込み容量を記録するものである請求項５記載のフラッシュメモリー記憶システム。
【請求項１４】
前記制御器が前記ホストシステムからフォーマット指令またはクリアー指令を受信する時、前記制御器が前記書き込み容量パラメーターを初期化するものである請求項１３記載のフラッシュメモリー記憶システム。
【請求項１５】
前記制御器が前記ホストシステムから書き込み指令を受信するとともに、前記論理ブロックの少なくとも一部にマップされた前記実体ブロックへ前記データを書き込んだ後、前記制御器が前記書き込み容量パラメーターを更新する請求項１３記載のフラッシュメモリー記憶システム。
【請求項１６】
前記制御器が前記ホストシステムから次の書き込み可能アドレス照会指令を受信した時、前記制御器は、前記書き込み容量パラメーターに従って次の書き込み可能アドレスを前記ホストシステムへ送り返すものである請求項１３記載のフラッシュメモリー記憶システム。
【請求項１７】
前記制御器が前記ホストシステムから読み出し指令または書き込み指令を受信した時、前記制御器が、変換パラメーターを使用して前記読み出し指令または前記書き込み指令により指示された論理アドレスおよびデータ容量を計算する請求項５記載のフラッシュメモリー記憶システム。
【請求項１８】
前記制御器が前記ホストシステムからエラー訂正データ照会指令を受信した時、前記制御器が、非支援メッセージを前記ホストシステムへ送り返すものである請求項５記載のフラッシュメモリー記憶システム。
【請求項１９】
前記制御器が前記ホストシステムからMSF読み出し指令を受信した時、前記制御器が、下記する数式（１）を介して前記MSF読み出し指令に対応する論理アドレスを計算するものであり、

LBA＝（（M×60＋S-2）×75＋F）×4…（１）

そのうち、LBAが前記フラッシュメモリーチップ中の前記MSF読み出し指令に対応する前記論理アドレスを表し、Mが前記MSF読み出し指令の分パラメーターを表し、Sが前記MSF読み出し指令の秒パラメーターを表し、Fが前記MSF読み出し指令のフレームパラメーターを表すものである、請求項５記載のフラッシュメモリー記憶システム。

【請求項２０】

前記制御器が前記ホストシステムからディスク排出指令を受信した時、前記制御器が、受信済みメッセージまたは準備済みメッセージを前記ホストシステムへ送り返すものである請求項５記載のフラッシュメモリー記憶システム。

【請求項２１】

前記ホストシステムが、ディスク装置の書き換え可能な蓄積媒体、コンパクトディスク読み出し専用メモリー（CD-ROM）装置の読み出し専用蓄積媒体または書き換え可能なディスク装置の書き換え可能な蓄積媒体を前記論理ブロックの別な部分として識別するものである請求項５記載のフラッシュメモリー記憶システム。

【請求項２２】

前記制御器が、指令に従って書き込み保護情報を設定するとともに、前記書き込み保護情報を前記実体ブロックの１つに書き込み、そのうち、前記書き込み保護情報が前記論理ブロックの前記一部が書き込み保護状態または非書き込み保護状態にあるかどうかを記録する請求項５記載のフラッシュメモリー記憶システム。

【請求項２３】

前記制御器が前記ホストシステムからクリアー指令、フォーマット指令またはセクション閉鎖指令を受信した時、前記制御器は、前記書き込み保護情報が前記書き込み保護状態を指示しているか否か判定し、

そのうち、前記書き込み保護情報が前記書き込み保護状態を指示している時、前記制御器が前記ホストシステムへエラーメッセージを送り返すものである請求項２２記載のフラッシュメモリー記憶システム。

【請求項２４】

前記制御器が前記ホストシステムから蓄積媒体特性照会指令を受信した時、前記制御器は、前記書き込み保護情報が前記書き込み保護状態を指示しているかどうか判定し、

そのうち、前記書き込み保護情報が前記書き込み保護状態を指示している時、前記制御器が前記ホストシステムへ単一層読み出し専用DVDメッセージまたは二重層読み出し専用DVDメッセージを送り返すものである請求項２２記載のフラッシュメモリー記憶システム。

【請求項２５】

フラッシュメモリー制御器であり、フラッシュメモリーチップの多数の実体ブロックを管理するためのものであって、前記フラッシュメモリー制御器が、

マイクロプロセッサーユニットと、

前記マイクロプロセッサーユニットに連結されるとともに、前記フラッシュメモリーチップに連結されるよう配列されるフラッシュメモリーインターフェースユニットと、

前記マイクロプロセッサーユニットに連結されるとともに、ホストシステムに連結されるよう配列されるホストインターフェースユニットと、

前記マイクロプロセッサーユニット連結されるとともに、多数の論理ブロックに配列され、かつ前記した多数の論理ブロックを前記した多数の実体ブロックの一部にマップするメモリー管理ユニットとを含み、

そのうち、前記メモリー管理ユニットが前記ホストシステムからの少なくとも１つの書き換え可能なディスク指令を識別するとともに、前記した少なくとも１つの書き換え可能なディスク指令に従って前記ホストシステムからのデータを前記した多数の論理ブロックの少なくとも一部にマップされた前記した多数の実体ブロックへ書き込むものである、フラッシュメモリー制御器。

【請求項２６】

前記メモリー管理ユニットが、書き換え可能なディスク装置の少なくとも１つの管理情

SAMSUNG EX1002, Page 444 of 1031

報を確立するとともに、前記書き換え可能なディスク装置の前記した少なくとも１つの管理情報を前記した多数の実体ブロックの１つへ保存するものである請求項２５記載のフラッシュメモリー制御器。

【請求項２７】

前記書き換え可能なディスク装置の前記した少なくとも１つの管理情報が蓄積状態情報を含むとともに、前記蓄積状態情報が前記した多数の論理ブロックの少なくとも１つの蓄積状態を記録するものである、請求項２６記載のフラッシュメモリー制御器。

【請求項２８】

前記書き換え可能なディスク装置の前記した少なくとも１つの管理情報が書き込み容量パラメーターを含むとともに、前記書き込み容量パラメーターが前記した多数の論理ブロックの少なくとも１つの書き込み容量を記録するものである、請求項２６記載のフラッシュメモリー制御器。

【請求項２９】

コンピューターシステムであって、

中央処理ユニット（CPU）と、

前記CPUに連結されたメインメモリーと、

前記CPUに連結された埋め込み式フラッシュメモリー記憶装置とを含み、そのうち、前記埋め込み式フラッシュメモリー記憶装置が、

多数の実体ブロックを有するフラッシュメモリーチップと、

前記フラッシュメモリーチップに連結されるとともに、多数の論理ブロックに配列され、かつ前記した多数の論理ブロックを前記した複数の実体ブロックの一部にマップする制御器とを含み、

そのうち、前記制御器が前記CPUから少なくとも１つの書き換え可能なディスク指令を識別するとともに、前記書き換え可能なディスク指令に従って、前記した多数の論理ブロックの少なくとも一部にマップされた前記した多数の実体ブロックへデータを書き込むものである、コンピューターシステム。

【請求項３０】

前記制御器が、書き換え可能なディスク装置の少なくとも１つの管理情報を確立するとともに、前記書き換え可能なディスク装置の前記した少なくとも１つの管理情報を前記した多数の実体ブロックの１つに蓄積するものである請求項２９記載のコンピューターシステム。

【請求項３１】

前記書き換え可能なディスク装置の前記した少なくとも１つの管理情報が、蓄積状態情報を含むとともに、前記蓄積状態情報が前記した多数の論理ブロックの少なくとも一部分の蓄積状態を記録するものである請求項３０記載のコンピューターシステム。

【請求項３２】

前記書き換え可能なディスク装置の前記した少なくとも１つの管理情報が、書き込み容量パラメーターを含むとともに、前記書き込み容量パラメーターが多数の論理ブロックの少なくとも一部分の書き込み容量を記録するものである請求項３０記載のコンピューターシステム。

【請求項３３】

コンピューターシステムであって、

CPUと、

前記CPUに連結されたメインメモリーと、

前記CPUに連結されたカードリーダーであり、そのうち、前記カードリーダーがカードリーダー制御器を含むとともに、前記カードリーダー制御器は、前記カードリーダーが書き換え可能なディスク装置であるとCPUへ宣言するよう配列されるものである、カードリーダーと、

前記カードリーダーにプラグされた取外し可能なメモリーカードとを含み、そのうち、前記取外し可能なメモリーカードが、

多数の実体ブロックを有するフラッシュメモリーチップと、
フラッシュメモリーチップに連結されるとともに、多数の論理ブロックに配列され、かつ前記した多数の論理ブロックを前記した複数の実体ブロックの一部にマップするメモリーカード制御器とを含み、
そのうち、前記カードリーダー制御器が前記CPUから少なくとも１つの書き換え可能なディスク指令を識別するとともに、前記書き換え可能なディスク指令に従って、前記した多数の論理ブロックの少なくとも一部にマップされた前記した多数の実体ブロックへデータを書き込む、コンピューターシステム。
【請求項３４】
フラッシュメモリー記憶システムであって、
カードリーダー制御器を含むカードリーダーであり、そのうち、前記カードリーダー制御器は、前記カードリーダーが書き換え可能なディスク装置であるとホストシステムへ宣言するよう配列されるものであるカードリーダーと、
前記カードリーダーにプラグされた取外し可能なメモリーカードとを含み、そのうち、前記取外し可能なメモリーカードが、
多数の実体ブロックを有するフラッシュメモリーチップと、
フラッシュメモリーチップに連結されるとともに、多数の論理ブロックに配列されるよう配列され、かつ前記した多数の論理ブロックを前記した複数の実体ブロックの一部にマップするメモリーカード制御器とを含み、
そのうち、前記カードリーダー制御器が前記ホストシステムから少なくとも１つの書き換え可能なディスク指令を識別するとともに、前記書き換え可能なディスク指令に従って、前記した多数の論理ブロックの少なくとも一部にマップされた前記した多数の実体ブロックへ前記ホストシステムからのデータを書き込む、フラッシュメモリー記憶システム。
【請求項３５】
フラッシュメモリー記憶装置を使用することにより書き換え可能なディスク装置をシミュレーションする方法であって、そのうち、前記フラッシュメモリー記憶装置が、フラッシュメモリーチップを有するとともに、前記フラッシュメモリーチップが多数の実体ブロックを有し、前記シミュレーション方法が、
多数の論理ブロックを配列するとともに、前記した多数の論理ブロックを前記した多数の実体ブロックの一部にマップするものであり、そのうち、前記した多数の論理ブロックが多数の論理アドレスを有し、前記した多数の実体ブロックが多数の実体アドレスを有するとともに、前記した多数の論理アドレスおよび前記した多数の実体アドレス間に存在する関係をマップすることと、
前記書き換え可能なディスク装置の少なくとも１つの管理情報を前記した多数の実体ブロックの１つに保存することと、
前記書き換え可能なディスク装置の前記した少なくとも１つの管理情報に従って、ホストシステムへ応答して前記フラッシュメモリー記憶装置へアクセスすることと、
前記ホストシステムにより与えられる少なくとも１つの前記書換可能なディスク指令の１つの書き込み指令に従って、変換パラメーターを使用して書き込み論理アドレスおよびデータ量を計算することと、を含むシミュレーション方法。
【請求項３６】
前記書き換え可能なディスク指令に従ってデータが前記した多数の実体ブロックの１つに書き込まれるとともに、書き込みエラーが発生した時、前記データが前記した多数の実体ブロックの別な１つに書き込まれるものである請求項３５記載のシミュレーション方法。
【請求項３７】
前記書き換え可能なディスク装置の前記した少なくとも１つの管理情報が蓄積状態情報を含むとともに、前記蓄積状態情報が前記した多数の論理ブロックの少なくとも一部の蓄積状態を記録するものである請求項３５記載のシミュレーション方法。
【請求項３８】

前記書き換え可能なディスク装置の前記した少なくとも１つの管理情報が書き込み容量パラメーターを含むとともに、前記書き込み容量パラメーターが前記した多数の論理ブロックの少なくとも一部の書き込み容量を記録するものである請求項３５記載のシミュレーション方法。

【発明の詳細な説明】

【技術分野】

【０００１】

この発明は、一般的にフラッシュメモリー記憶システムに関し、特に、書き換え可能ディスク装置（Rewritable Disc Device）をシミュレーションするためのフラッシュメモリー記憶システム、フラッシュメモリー制御器、そのフラッシュメモリー記憶システムを装備したコンピューターシステムおよび書き換え可能ディスクをシミュレーションする方法に関する。

【背景技術】

【０００２】

書き換え可能なデジタル多用途ディスク（DVD = Digital Versatile Disc)ドライブは、光学ディスク記憶装置である。DVDは、その大容量のために、高画質ビデオデータ、高品質オーディオデータおよび大量データを保存するため、次第に従来のCD（Compact Disc）に置き換えられてきている。DVDは、DVDドライブの光学ピックアップを介してアクセスされなければならない。しかし、DVDドライブ2002（図１Ａに図示）の大きな体積および高い電力消費の故に、スモールコンピューター（例えば、ネットブック）は、通常、いかなるDVDドライブも装備できず、ユーザーにとって非常に不便である。また、DVDドライブ2002が機械的光学ピックアップを使用して、DVD 2004からデータ読み出し又はDVD 2004へデータを焼き付けるので、DVDドライブ2002のアクセス速度が非常に遅い。さらに、DVD 2004の体積が大きく、持ち運びに不便である。

【０００３】

上記の問題は、もしもDVDドライブ2002がフラッシュメモリー（Flash Memory）を有するフラッシュドライブ2006に置き換えられるならば、フラッシュメモリーのデータ非揮発性、低い電力消費、小さい体積および非機械的構造を蓄積媒体（storage medium）とする故に、解決可能である。

【０００４】

フラッシュメモリーをその蓄積媒体として使用するユニバーサルシリアルバス（Universal Serial Bus = USB）記憶装置が特許文献１に開示されており、そのうち、USB記憶装置がCD-ROM装置をシミュレートして、USB記憶装置に保存されているデータがアクシデントによって削除されることを防止できる。しかし、ユーザーは、USB記憶装置によりシミュレートされたCD-ROM装置を使用することによりデータを書き換えられないので、データ書き換え性の要求（requirement of data rewritability）を満足させることができない。

【先行技術文献】

【特許文献】

【０００５】

【特許文献１】米国特許第７，１１１，１２１号明細書

【発明の概要】

【発明が解決しようとする課題】

【０００６】

そこで、この発明の目的は、書き換え可能ディスク装置をシミュレートできるフラッシュメモリー記憶システムを提供することにある。

【０００７】

この発明の目的は、フラッシュメモリーチップを使用することにより書き換え可能蓄積媒体をシミュレーションできるフラッシュメモリー制御器を提供することにある。

【０００８】

この発明の目的は、書き換え可能ディスク指令に従って、フラッシュメモリー記憶装置にアクセスできるコンピューターシステムを提供することにある。

【０００９】

この発明の目的は、フラッシュメモリー記憶装置を使用することによって、書き換え可能ディスクをシミュレーションできる方法を提供することにある。

【課題を解決するための手段】

【００１０】

この発明の実施形態がフラッシュメモリーチップとコネクターと制御器とを含むフラッシュメモリー記憶システムを提供する。フラッシュメモリーチップが多数の実体ブロックを有する。コネクターがホストシステムに連結されるよう配列される。制御器がフラッシュメモリーチップおよびコネクターに連結される。制御器が多数の論理ブロックを配列するとともに、論理ブロックを実体ブロックの一部にマップする（map 「写像する」とも言う）。また、制御器が更にホストシステムからの少なくとも１つの書き換え可能ディスク指令を識別とともに、少なくとも１つの書き換え可能ディスク指令に従ってホストシステムからのデータを論理ブロックの少なくとも一部にマップされたへ実体ブロックに書き込む。

【００１１】

この発明がフラッシュメモリーチップとコネクターと制御器とを含むフラッシュメモリー記憶システムを提供する。フラッシュメモリーチップが多数の実体ブロックを有する。コネクターがホストシステムに連結されるよう配列される。制御器がフラッシュメモリーチップおよびコネクターに連結される。制御器が多数の論理ブロックを配列するとともに、論理ブロックを実体ブロックの一部にマップする。実体ブロックの１つが書き換え可能なディスク装置の少なくとも１つの管理情報を保存する。制御器および前記書き換え可能なディスク装置の少なくとも１つの管理情報を配列することにより、フラッシュメモリー記憶システムがホストに連結される時、ホストが論理ブロックの少なくとも一部を書き換え可能なディスク装置の書き換え可能な蓄積媒体として識別することを許すものである。

【００１２】

この発明は、フラッシュメモリーチップの多数の実体ブロックを管理するためのフラッシュメモリー制御器を提供する。フラッシュメモリー制御器がマイクロプロセッサーユニットとフラッシュメモリーインターフェースユニットとホストインターフェースユニットとメモリー管理ユニットとを含む。フラッシュメモリーインターフェースユニットがマイクロプロセッサーユニットに連結されるとともに、フラッシュメモリーチップに連結されるよう配列される。ホストインターフェースユニットがマイクロプロセッサーユニットに連結されるとともに、ホストシステムに連結されるよう配列される。メモリー管理ユニットがマイクロプロセッサーユニット連結される。メモリー管理ユニットが多数の論理ブロックに配列するとともに、論理ブロックを実体ブロックの一部にマップする。また、メモリー管理ユニットがホストシステムからの少なくとも１つの書き換え可能なディスク指令を識別するとともに、少なくとも１つの書き換え可能なディスク指令に従ってホストシステムからのデータを論理ブロックの少なくとも一部にマップされた実体ブロックへ書き込むものである。

【００１３】

この発明は、中央処理ユニット（CPU）とメインメモリーと埋め込み式フラッシュメモリー記憶装置とを含むコンピューターシステムを提供する。埋め込み式フラッシュメモリー記憶装置がCPUに連結されるとともに、フラッシュメモリーチップと制御器とを含み、そのうち、フラッシュメモリーチップが多数の実体ブロックを有し、かつ制御器がフラッシュメモリーチップに連結される。制御器が多数の論理ブロックに配列され、かつ論理ブロックを実体ブロックの一部にマップする。また、制御器がCPUから少なくとも１つの書き換え可能なディスク指令を識別するとともに、書き換え可能なディスク指令に従って、論理ブロックの少なくとも一部にマップされた実体ブロックへデータを書き込むものである。

【００１４】
この発明は、CPUとメインメモリーとカードリーダーと取外し可能なメモリーカードとを含むコンピューターシステムを提供する。カードリーダーがCPUに連結されるとともに、カードリーダー制御器を有し、そのうち、カードリーダー制御器がカードリーダーが書き換え可能なディスク装置であるとCPUへ宣言する。取外し可能なメモリーカードがカードリーダーにプラグされる。取外し可能なメモリーカードがフラッシュメモリーチップとメモリーカード制御器とを含み、そのうち、フラッシュメモリーチップが多数の実体ブロックを有し、かつメモリーカード制御器がフラッシュメモリーチップに連結される。メモリーカード制御器が多数の論理ブロックに配列されるとともに、多数の論理ブロックを実体ブロックの一部にマップする。また、メモリーカード制御器がCPUから少なくとも１つの書き換え可能なディスク指令を識別するとともに、書き換え可能なディスク指令に従って、論理ブロックの少なくとも一部にマップされた実体ブロックへデータを書き込む。
【００１５】
この発明は、カードリーダーと前記取外し可能なメモリーカードとを含むフラッシュメモリー記憶システムを提供する。カードリーダーがカードリーダー制御器を含むとともに、カードリーダー制御器がカードリーダーが書き換え可能なディスク装置であるとホストシステムへ宣言するよう配列される。取外し可能なメモリーカードがカードリーダーにプラグされる。取外し可能なメモリーカードがフラッシュメモリーチップとメモリーカード制御器とを含み、そのうち、フラッシュメモリーチップが多数の実体ブロックを有するとともに、メモリーカード制御器がフラッシュメモリーチップに連結される。メモリーカード制御器が多数の論理ブロックを配列するとともに、論理ブロックを実体ブロックの一部にマップする。カードリーダー制御器がホストシステムから少なくとも１つの書き換え可能なディスク指令を識別するとともに、書き換え可能なディスク指令に従って、論理ブロックの少なくとも一部にマップされた実体ブロックへホストシステムからのデータを書き込む。
【００１６】
この発明は、フラッシュメモリー記憶装置を使用することにより書き換え可能なディスク装置をシミュレーションする方法を提供する。フラッシュメモリー記憶装置がフラッシュメモリーチップを有し、そのうち、フラッシュメモリーチップが多数の実体ブロックを有する。この方法は、多数の論理ブロックを配列するとともに、論理ブロックを実体ブロックの一部にマップするものであり、そのうち、論理ブロックが多数の論理アドレスを有し、実体ブロックが多数の実体アドレスを有するとともに、論理アドレスおよび実体アドレス間に存在する関係をマップすることを含む。この発明は、また、書き換え可能なディスク装置の少なくとも１つの管理情報に従って、ホストシステムへ応答してフラッシュメモリー記憶装置へアクセスすることを含む。この発明は、更に、ホストシステムにより与えられる少なくとも１つの前記書換可能なディスク指令の１つの書き込み指令に従って、変換パラメーターを使用して書き込み論理アドレスおよびデータ量を計算することを含み、そのうち、書き込み論理アドレスが論理アドレスの１つである。
【００１７】
上記したように、この発明の実施形態に従ったフラッシュメモリー記憶装置、フラッシュメモリー制御器、コンピューターシステムおよび方法は、フラッシュメモリーチップを使用することにより書き換え可能なディスクをシミュレーションできる。
【発明の効果】
【００１８】
つまり、この発明の実施形態中、書き換え可能なDVD装置は、フラッシュメモリー記憶装置を使用することによりシミュレーションされるとともに、様々なデータ操作が様々な書き換え可能なDVD指令に従って、シミュレーションされた書き換え可能なDVD装置上で実施される。フラッシュメモリー記憶システムのアクセス速度が一般的な書き換え可能なDVD装置のそれよりも速いので、フラッシュメモリー記憶装置を使用することによりシミュレーションされた書き換え可能なDVD装置は、一般的な書き換え可能なDVD装置よりも速い

アクセス速度を有する。また、フラッシュメモリー記憶装置を使用することによりシミュレーションされた書き換え可能なDVD装置は、いかなる機械的な光学ピックアップも必要としない。従って、書き換え可能なDVD装置のサイズが大幅に削減される。それにより、フラッシュメモリー記憶装置を使用することによりシミュレーションされた書き換え可能なDVD装置は、小型ネットブック中に配置できる。

【図面の簡単な説明】

【００１９】

【図１Ａ】従来の書き換え可能ディスク装置およびフラッシュドライブを示す図である。

【図１Ｂ】この発明の第１の実施形態にかかるホストシステムおよびフラッシュメモリー記憶装置を示すブロック図である。

【図１Ｃ】この発明の実施形態にかかるコンピューター、入力／出力装置およびフラッシュメモリー記憶装置を示す説明図である。

【図２】この発明の実施形態にかかるフラッシュメモリーチップを示すブロック説明図である。

【図３】この発明の実施形態にかかる論理ブロックの構成を示す説明図である。

【図４】この発明の実施形態にかかる書き換え可能なデジタル多用途ディスク（ＤＶＤ）をシミュレーションする方法を示すフローチャート（流れ図）である。

【図５】この発明の第２の実施形態にかかるフラッシュメモリー記憶システムを示すブロック図である。

【発明を実施するための形態】

【００２０】

以下、この発明を実施するための形態を図面に基づいて説明する。

一般的に言って、フラッシュメモリー記憶装置（フラッシュメモリー記憶システムとも言う）は、フラッシュメモリーチップと制御器（制御回路とも言う）とを含む。フラッシュメモリー記憶装置は、ホストシステムと一緒に使用されて、ホストシステムがフラッシュメモリー記憶装置へデータを書き込んだりフラッシュメモリー記憶装置からデータを読み出したりできる。加えて、フラッシュメモリー記憶装置は、また、嵌め込み式（embeded）フラッシュメモリーと、ホストシステム上で実質的に実行されて嵌め込み式フラッシュメモリーの制御器として供されるソフトウェアとを含み得る。

【００２１】

図１Ｂは、この発明の第１の実施形態にかかるホストシステムおよびフラッシュメモリー記憶装置を示すブロック図である。

【００２２】

図１Ｂにおいて、ホストシステム1000がコンピューター1100と入力／出力（input/output=I/O）装置1106とを含む。

【００２３】

コンピューター1100が中央処理ユニット（CPU）1102とメインメモリー1104とシステムバス1108とデータ伝送インターフェース1110とメインディスクドライブ1112とを含む。コンピューター1100がディスクトップコンピューターまたはノートブックコンピューターで有り得る。

【００２４】

CPU 1102がマンドを解釈するとともに、主要な操作を実行する。メインメモリー1104が一時的にデータおよびプログラムを保存するために配列される。システムバス1108がホストシステム1000の多様なコンポーネントに接続するために配列される。データ伝送インターフェース1110が様々な周辺機器に接続するために配列される。メインディスクドライブ1112が操作システム（Operation System = OS）1122および焼き付けプログラム1124をインストールするために配置される。OS 1122がホストシステム1000のハードウェアおよびソフトウェアを管理するために配列される。この実施形態中、OS 1122がMicrosoft Windows（登録商標）7であり得る。しかし、この発明は、それに限定されるものではない。焼き付けプログラム1124は、ユーザーがデータをフラッシュメモリー記憶装置100に焼き付

ける又は同者（same）からデータを消去するようなユーザーインターフェースを提供するために配列される。

【００２５】

図１Ｃに示すように、I/O 装置1106がマウス1202とキーボード1204とディスプレイ1206とプリンター1208とを含む。注意すべきは、図１Ｃに示めされたデバイスは、I/O 装置1106を限定しようというものではないとともに、I/O 装置1106が更に他のデバイスを含み得ることである。

【００２６】

この実施形態中、フラッシュメモリー記憶装置100は、データ伝送インターフェース1110を介してホストシステム1000の他のコンポーネントに連結される。ホストシステム1000は、CPU 1102とメインメモリー1104とI/O 装置1106とメインディスクドライブ1112との操作を介してフラッシュメモリー記憶装置100へデータを書き込む又はフラッシュメモリー記憶装置100からデータを読み出すことができる。フラッシュメモリー記憶装置100は、図１Ｃに示すように、フラッシュドライブ1212、メモリーカード1214、固態ドライバー（Solid State Driver = SSD）1216または嵌め込み式記憶装置1218であることができる。注意すべきことは、もしフラッシュメモリー記憶装置100が嵌め込み式記憶装置（即ち、嵌め込み式記憶装置1218）として履行されるならば、嵌め込み式フラッシュメモリー記憶装置100は、ホストシステム1000のマザーボード（図示せず）に直接嵌め込まれることである。

【００２７】

フラッシュメモリー記憶装置100がコネクター102とフラッシュメモリー制御器104とフラッシュメモリーチップ106とを含む。

【００２８】

コネクター102がフラッシュメモリー制御器104に連結されるとともに、ホストシステム1000に連結されるよう配列される。この実施形態中、コネクター102がユニバーサルシリアルバス（Universal Serial Bus = USB）コネクターである。しかし、この発明は、それに限定されるものではないとともに、コネクター102は、また、パラレルアドバンスドテクノロジーアタッチメント（Parallel Advanced Technology Attachment = PATA）コネクター、シリアルアドバンスドテクノロジーアタッチメント（Serial Advanced Technology Attachment = SATA）コネクター、IEEE（Institute of Electrical and Electronics Engineers）1394コネクター、周辺部品相互連絡（Peripheral Component Interconnect = PCI）高速コネクター、安全デジタル（Secure Digital）インターフェースコネクター、メモリースティック（Memory Stick = MS）インターフェースコネクター、マルチメディアカード（Multi Media Card = MMC）インターフェースコネクター、コンパクトフラッシュ（登録商標）（Compact Flash = CF）インターフェースコネクター、総合デバイス電子（Integrated Device Electronic = IDE）コネクターまたは他の適合するコネクターで有り得る。

【００２９】

フラッシュメモリー制御器104は、ハードウェアまたはファームウェア（firmware）の形式で履行された多数の論理ゲート、あるいは制御指令を実行するために配列されるとともに、ホストシステム1000の指令に従ってフラッシュメモリーチップ106で様々なデータ操作を実施する。フラッシュメモリー制御器104がマイクロプロセッサーユニット202とホストインターフェースユニット206とフラッシュメモリーインターフェースユニット208を含む。

【００３０】

マイクロプロセッサーユニット202は、フラッシュメモリー制御器104のメイン制御ユニットであるとともに、それがメモリー管理ユニット204とホストインターフェースユニット206とフラッシュメモリーインターフェースユニット208と合作してフラッシュメモリー記憶装置100の様々な操作を実行する。

【００３１】

メモリー管理ユニット204がマイクロプロセッサーユニット202に連結されるとともに、この実施形態に従って、データ書き込みメカニズムおよびブロック管理メカニズムを実行するために配列される。とりわけ、メモリー管理ユニット204は、フラッシュメモリー記憶装置100が書き換え可能なデジタル多用途ディスク（Digital Versatile Disc = DVD)装置であるとホストシステム1000に宣言し、書き換え可能なDVD装置の管理情報を確立するとともに、管理情報に基づいてフラッシュメモリー記憶装置100へアクセスするようホストシステム1000へ応答する。

【００３２】

例えば、メモリー管理ユニット204は、フラッシュメモリー記憶装置100がDVD+RW規格、DVD-RW規格、DVD-RAM規格またはBlu-Ray Disc(BD)規格に適合するディスク装置であるとホストシステム1000に対して宣言する。その後、メモリー管理ユニット204がホストシステム1000から受信した書き換え可能なDVD指令を識別し、かつ書き換え可能なDVD指令に従ってフラッシュメモリーチップ106を操作する。例えば、フラッシュメモリーチップ106がホストシステム1000から受信したデータを書き換え可能なDVD指令に従ってフラッシュメモリーチップ106へ焼き付ける。つまり、ホストシステム1000のユーザーが書き換え可能なDVD装置の焼き付けプログラムを介してフラッシュメモリー記憶装置100を操作する時、メモリー管理ユニット204が焼き付け指令を識別し、かつ焼き付け指令に従ってフラッシュメモリーチップ106上で様々なデータ操作を実施する。以下、メモリー管理ユニット204の操作が添付図面とともに詳述される。

【００３３】

この実施形態中、メモリー管理ユニット204がファームウェア形式（firmware form）でフラッシュメモリー制御器104中で履行される。例えば、メモリー管理ユニット204がプログラムメモリー（例えば、Read Only Memory = ROM）に焼き付けられた多数の制御指令を含むとともに、プログラムメモリーがメモリー管理ユニット204中に嵌め込まれる。フラッシュメモリー記憶装置100が操作中である時、メモリー管理ユニット204の制御指令がマイクロプロセッサーユニット202によって実行され、この発明の実施形態に従って、焼き付けメカニズム（例えば、データの読み出し、書き込み、消去）およびフラッシュメモリー管理メカニズムを達成する。

【００３４】

この発明の他の実施形態中、メモリー管理ユニット204の制御指令がプログラムコードとして特定領域（例えば、システムデータを専門に保存するためのフラッシュメモリーチップ中のシステム領域）に保存される。同様に、メモリー管理ユニット204の制御指令は、フラッシュメモリー記憶装置100が操作中である時、マイクロプロセッサーユニット202によって実行される。また、更に他の実施形態中、メモリー管理ユニット204がまたファームウェアとしてフラッシュメモリー制御器104中で履行される。

【００３５】

ホストインターフェースユニット206がマイクロプロセッサーユニット202に連結されるとともに、ホストシステム1000からの指令およびデータを受信かつ識別する。つまり、ホストシステム1000から受信した指令およびデータは、ホストインターフェースユニット206を介してマイクロプロセッサーユニット202へ伝送される。この実施形態中、ホストインターフェースユニット206は、コネクター102に対応するUSBインターフェースである。しかし、この発明は、それに限定されるものではないとともに、ホストインターフェースユニット206は、PATAインターフェース、SATAインターフェース、IEEE 1394インターフェース、PCI高速インターフェース、SDインターフェース、MSインターフェース、MMCインターフェース、CFインターフェース、IEDインターフェースまたは他の適合する伝送インターフェースで有り得る。

【００３６】

フラッシュメモリーインターフェースユニット208がマイクロプロセッサーユニット202に連結され、かつフラッシュメモリーチップ106にアクセスするよう配列される。即ち、フラッシュメモリーチップ106へ書き込まれるデータは、フラッシュメモリーインターフ

ェースユニット208によりフラッシュメモリーチップ106が受け入れることのできるフォーマットに変換される。

【００３７】

この発明の実施形態中、フラッシュメモリー制御器104が更にバッファーメモリー252と電源管理ユニット254とエラー訂正ユニット256とを含む。

【００３８】

バッファーメモリー252がマイクロプロセッサーユニット202に連結されるとともに、ホストシステム1000から受信したデータおよび指令またはフラッシュメモリーチップ106から受信したデータを一時的に保存する。

【００３９】

電源管理ユニット254がマイクロプロセッサーユニット202に連結され、かつフラッシュメモリー記憶装置100の電源供給を制御するよう配列される。

【００４０】

エラー訂正ユニット256がマイクロプロセッサーユニット202に連結されるとともに、エラー訂正手続を実行するよう配列されてデータの正確性を確保する。具体的には、メモリー管理ユニット204がホストシステム1000から書き込み指令を受信した時、エラー訂正ユニット256が書き込み指令に対応するデータのためにエラー検査訂正（Error Checking and Correcting = ECC）コードを発生させるとともに、メモリー管理ユニット204がデータおよび対応するECCコードを共にフラッシュメモリーチップ106へ書き込む。続いて、メモリー管理ユニット204がフラッシュメモリーチップ106からデータを読み出す時、それは、データに対応するECCコードも読み出すとともに、エラー訂正ユニット256がECCコードに従ってエラー訂正手続を実行する。

【００４１】

フラッシュメモリーチップ106がフラッシュメモリー制御器104に連結され、かつデータを保存するよう配列される。フラッシュメモリーチップ106が多数の実体ブロックを有する。フラッシュメモリー中、各実体ブロックは、最小消去ユニットである。つまり、各実体ブロックが一緒に消去される最小数量のメモリーセルを含む。各実体ブロックが多数の実体アドレスを有する。この実施形態中、実体アドレスは、実体ページである。しかし、この発明は、それに限定されない。各実体ページは、最小プログラムユニットである。即ち、各実体ページは、データを書き込み又は読み出すための最小ユニットである。各実体ページは、通常、ユーザーデータ領域および冗長領域（redundant area）を含む。ユーザーデータ領域がユーザーデータを保存する一方、冗長領域がシステムデータ（例えば、ECCコード）を保存する。この実施形態中、フラッシュメモリーチップ106は、マルチレベルセル（Multi Level Cell = MLC）NANDフラッシュメモリーチップである。しかし、この発明は、それに限定されないとともに、フラッシュメモリーチップ106がシングルレベルセル（Single Level Cell = SLC）NANDフラッシュメモリーチップでも有り得る。

【００４２】

図２は、この発明の実施形態にかかるフラッシュメモリーチップを示すブロック説明図である。

【００４３】

図２において、フラッシュメモリーチップ106が実体ブロックPBA(0)-PBA(N)を有する。各実体ブロックが最小消去ユニットである。つまり、各実体ブロックが一緒に消去される最小数量のメモリーセルを含む。各実体ブロックが128の実体アドレス（即ち、実体ページ）を有する。しかし、この発明は、それに限定されないとともに、この発明の他の実施形態中、各実体ブロックが64，256または他の適合する数量の実体アドレスを有することもあり得る。

【００４４】

メモリー管理ユニット204が実体ブロックPBA(0)-PBA(N)を論理的にシステム領域（system area）302と蓄積領域（storage area）304と交替領域（replacement area）306とに分類する。

【００４５】
論理的にシステム領域302に属する実体ブロックPBA(0)-PBA(S)が、フラッシュメモリーの製造業者および型番、各実体ブロックに対応する実体アドレスの数量のような、システムデータを記録するために使用される。
【００４６】
論理的に蓄積領域304（図２中央に符号）に属する実体ブロックPBA(S+1)-PBA(A)が、ホストシステム1000によって書き込まれたデータを保存するために使用される。つまり、フラッシュメモリー記憶装置100は、蓄積領域304中の実体ブロックを使用してホストシステム1000から受信したデータを実際に保存する。具体的には、蓄積領域304中の実体ブロックPBA(S+1)-PBA(A)が更にデータ領域304aと予備領域304bとに分類され、そのうち、データ領域304a中の実体ブロックPBA(S+1)-PBA(D)がデータを包含する実体ブロックであり、予備領域304b中の実体ブロックPBA(D+1)-PBA(A)がまだデータの保存に使用されていない実体ブロックである。さらに詳細には、メモリー管理ユニット204が予備領域304bから実体ブロックを得てデータを得られた実体ブロックへ書き込む時、得られた実体ブロックがデータ領域304aに連合され、かつデータ領域304aに連合された実体ブロックが消去された時、消去された実体ブロックは、予備領域304bに連合される。
【００４７】
論理的に交替領域306に属する実体ブロックPBA(A+1)-PBA(N)は、交替実体ブロックである。例えば、フラッシュメモリーチップ106が製造された時、その４％の実体ブロックが交替の目的で確保（reserved）されている。つまり、システム領域302および蓄積領域304中の実体ブロックが損傷した時、交替領域306中に確保された実体ブロックは、損傷した実体ブロック（即ち、悪いブロック）と置き換えるために使用できる。このように、もし未だ交替領域306中に正常な実体ブロックがあるとともに、実体ブロックが損傷された場合、メモリー管理ユニット204が交替領域306から正常な実体ブロックを手に入れて損傷した実体ブロックと置き換える。もしも交替領域306中に正常な実体ブロックがなく、かつ実体ブロックが損傷された場合、フラッシュメモリー記憶装置100は、書き込み保護（write protected）状態と宣言されるとともに、書き込みデータとして使用することができない。
【００４８】
注意すべきは、フラッシュメモリー記憶装置100の操作中、実体ブロックPBA(0)-PBA(N)がシステム領域302、データ領域304a、予備領域304bおよび交替領域306に分類されるという分類関係は、ダイナミックに変化することである。つまり、メモリー管理ユニット204が元は予備領域304bに属する１つの実体ブロック（例えば、実体ブロックPBA(D+2)）にデータを書き込む時、実体ブロックがデータ領域304aに連合される。あるいは、データ領域304a（又は予備領域304b）中の実体ブロックが損傷されたため、交替領域306から得られた実体ブロックによって置き換えられる時、元は交替領域306に属する実体ブロックがデータ領域304a（又は予備領域304b）に連合される。
【００４９】
フラッシュメモリー記憶装置100が蓄積領域304中の実体ブロックをかわるがわる使用してホストシステム1000から受信したデータを保存するので、メモリー管理ユニット204は、論理ブロックLBA(0)-LBA(L)を提供してホストシステム1000によりアクセスさせる。具体的には、各論理ブロックが多数の論理アドレスを含むとともに、各論理アドレスが実体ブロックの実体アドレスにマップされる。従って、ホストシステム1000のOS 1122が論理アドレスに従って単純にデータにアクセスする一方、メモリー管理ユニット204が対応する実体アドレス中のデータに実際にアクセスする。この実施形態中、メモリー管理ユニット204が論理ブロックLBA(0)-LBA(L)を蓄積領域304中の実体ブロックにマップするとともに、論理ブロック-実体ブロックマップテーブル（logical block-physical block mapping table）中に論理ブロックおよび実体ブロック間のマップ関係を記録する。この実施形態中、ホストシステム1000により書き込まれるデータが蓄積領域304に属する実体ブロックへ書き込まれるとともに、ホストシステム1000は、システム領域302および交替領域306

中の実体ブロックにアクセスできない。

【００５０】

図３は、この発明の実施形態にかかるロジックブロックの構成を示す説明図である。

【００５１】

図３において、メモリー管理ユニット204が論理ブロックLBA(0)-LBA(L)を第１論理ユニット360と第２論理ユニット380とに分類し、第１論理ユニット360が論理ブロックLBA(0)-LBA(G)を含み、第２論理ユニット380が論理ブロックLBA(G+1)-LBA(L)を含む。

【００５２】

この実施形態中、メモリー管理ユニット204が第１論理ユニット360を書き換え可能なDVDディスク（例えば、DVD+RW規格、DVD-RW規格、DVD-RAM規格またはBlue-Ray Disc(BD)規格に適合するディスク装置）として配列するとともに、第２論理ユニット380を大容量記憶装置（例えば、ファイル配置テーブル32（File Allocation Table 32 = FAT32）フォーマットまたはNTFSフォーマットに適合する大容量記憶装置）として配列する。

【００５３】

とりわけ、この実施形態中、フラッシュメモリー記憶装置100がホストシステム1000に連結されるとともに、ホストシステム1000のCPU 1102が、フラッシュメモリー記憶装置100が書き換え可能なDVD装置および大容量記憶装置からなると識別する。

【００５４】

具体的には、ホストシステム1000のCPU 1102がフラッシュメモリー記憶装置100からの装置特徴を要請する時、メモリー管理ユニット204は、フラッシュメモリー記憶装置100が大容量記憶カテゴリーであるとともに、第１論理ユニット360と第２論理ユニット380とを有することをホストシステム1000に宣言する。その後、CPU 1102がフラッシュメモリー記憶装置100から第１論理ユニット360の装置特徴を要請する（例えば、CPU 1102が0×12指令を介して第１論理ユニット360の装置特徴について質問する）時、メモリー管理ユニット204は、第１論理ユニット360が書き換え可能なDVD装置であるとホストシステム1000に宣言する（例えば、メモリー管理ユニット204がホストシステム1000に0×05メッセージで応答する）とともに、CPU 1102がフラッシュメモリー記憶装置100から第２論理ユニット380の装置特徴を要請する時、メモリー管理ユニット204は、第２論理ユニット380が大容量記憶装置であると宣言する（例えば、メモリー管理ユニット204がホストシステム1000に0×00メッセージで応答する）。従って、ホストシステム1000がフラッシュメモリー記憶装置100の書き換え可能なDVD装置（即ち、第１論理ユニット360）に焼き付けようとする時、メモリー管理ユニット204は、ホストシステム1000から受信したディスク指令に従って、ホストシステム1000からのデータを第１論理ユニット360の論理ブロックLBA(0)-LBA(G)にマップされた実体ブロックへ書き込む。また、ホストシステム1000がフラッシュメモリー記憶装置100の大容量記憶装置（即ち、第２論理ユニット380）に保存しょうとする時、ホストシステム1000から受信したディスク指令に従って、ホストシステム1000からのデータを第２論理ユニット380の論理ブロックLBA(G+1)-LBA(L)にマップされた実体ブロックへ書き込む。

【００５５】

注意すべきは、この実施形態中、たとえメモリー管理ユニット204が論理ブロックLBA(0)-LBA(L)を書き換え可能なDVDフォーマットを有する第１論理ユニット360およびディスクフォーマットを有する第２論理ユニット380へ分類するとしても、この発明は、それに限定されるものではないことである。例えば、この発明の他の実施形態中、第２論理ユニット380は、また、コンパクトディスク読み出し専用メモリー（Compact Disk-Read Only Memory = DC-ROM）として配列され得る。あるいは、この発明の更に別の実施形態中、第２論理ユニット380は、また、他の書き換え可能なDVD装置であり得るとともに、メモリー管理ユニット204は、第２論理ユニット380が書き換え可能なDVD装置（例えば、例えば、DVD+RW規格、DVD-RW規格、DVD-RAM規格またはBD規格に適合するディスク装置）であるとホストシステム1000に宣言するとともに、書き換え可能なDVD指令に従って第２論理ユニット380中のデータにアクセスする。更に、この発明のなお別な実施形態中、メモリー管理ユニ

ット204は、また、全ての論理ブロックLBA(0)-LBA(L)を書き換え可能なDVD装置として配列する、あるいは、論理ブロックLBA(0)-LBA(L)をより多くの論理ユニットに分類するとともに、これらの論理ブロックをそれぞれ書き換え可能なDVD装置、大容量記憶装置またはコンパクトディスク読み出し専用メモリー装置として配列する。

【００５６】

ここで、フラッシュメモリー制御器104のメモリー管理ユニット204が通常操作パターンに従って、ホストシステム1000からのディスク指令を実行して、第２論理ユニット380上で様々なデータ操作を実施する。また、フラッシュメモリー記憶装置100が第１論理ユニット360に与えられた書き換え可能なDVD指令を受信する時、メモリー管理ユニット204が適切なメッセージに応答して第１論理ユニット360を使用する書き換え可能なDVDをシミュレートする。以下、メモリー管理ユニット204の操作メカニズムが詳細に記述される。

【００５７】

この発明の実施形態中、ホストシステム1000のCPU 1102が蓄積媒体特徴照会指令（例えば、書き換え可能なDVD指令の0×46指令）をフラッシュメモリー記憶装置100へ送信して第１論理ユニット360の蓄積媒体特徴を入手する時、メモリー管理ユニット204は、単一層書き換え可能DVDメッセージをホストシステム1000へ送信する。従って、CPU 1102が第１論理ユニット360の蓄積媒体を単一層（single layer）書き換え可能DVDとして識別する。注意すべきことは、この発明の他の実施形態中、メモリー管理ユニット204が第１論理ユニット360の容量に従って対応する蓄積媒体特徴を送信することもあり得ることである。例えば、もし第１論理ユニット360の容量が4.7GBより小さいか等しい場合、メモリー管理ユニット204が単一層書き換え可能DVDメッセージをホストシステム1000へ送信する。もしも第１論理ユニット360の容量が8.5GBより大きいか等しい場合、メモリー管理ユニット204が二重層（double layer）書き換え可能DVDメッセージをホストシステム1000へ送信する。

【００５８】

この発明の実施形態中、ホストシステム1000のCPU 1102が容量照会指令（例えば、書き換え可能なDVD指令の0×25指令）をフラッシュメモリー記憶装置100へ送信して第１論理ユニット360の容量を入手する時、メモリー管理ユニット204は、第１論理ユニット360の容量値をホストシステム1000へ送り返す。例えば、もし総容量が15GBであれば、メモリー管理ユニット204は、第１論理ユニット360の容量が15GBであるとホストシステム1000に宣言する。注意すべきことは、この発明の他の実施形態中、メモリー管理ユニット204は、総容量がサポート（支持）される時、書き換え可能なDVDの設計仕様に従ってホストシステム1000へ応答することもあり得ることである。例えば、メモリー管理ユニット204は、もし総容量が15GBであれば、第１論理ユニット360の容量が4.7GBであるとホストシステム1000に宣言するとともに、メモリー管理ユニット204は、第１論理ユニット360が単一層書き換え可能DVDであると宣言する。さらに、メモリー管理ユニット204は、もし総容量が15GBであれば、第１論理ユニット360の容量が8.5GBであるとホストシステム1000に宣言するとともに、メモリー管理ユニット204は、第１論理ユニット360が二重層書き換え可能DVDであると宣言する。

【００５９】

この発明の実施形態中、メモリー管理ユニット204は、書き換え可能DVDブランクフラッグ（blank flag）を確立する。ここで、書き換え可能DVDブランクフラッグは、書き換え可能DVD装置の管理情報の１つであり、かつ第１論理ユニット360の蓄積状態を記録する。例えば、もしもフラッシュメモリー記憶装置100が製造されたばかりで何のデータも書き込まれていない場合、書き換え可能DVDブランクフラッグは、「ブランク状態」（blank status）として標示される。もしデータが既にフラッシュメモリー記憶装置100の第１論理ユニット360へ書き込まれていれば、書き換え可能DVDブランクフラッグは、「非ブランク状態」（non-blank status）として標示される。この実施形態中、書き換え可能DVDブランクフラッグ中の「0」が「ブランク状態」を表す一方、「1」が「非ブランク状態」を表す。詳しくは、書き換え可能DVDの同一蓄積アドレスにデータが再焼き付けされる時、書

き換え可能DVD全体が先ず消去されなければならない。かくして、ホストシステム1000がクリアー指令（例えば、書き換え可能DVD 指令の0×A1指令）を使用して第１論理ユニット360をクリアーする。しかし、フラッシュメモリーチップ106中の実体ブロックのユニットにおいてデータが消去されるため、データがその実体ブロックへ書き込まれる時にだけ、実体ブロックが消去される。CPU 1102がクリアー指令をフラッシュメモリー記憶装置100へ送信する時、メモリー管理ユニット204は、「ブランク状態」として書き換え可能DVDブランクフラッグを標示するとともに、クリアー指令が既に実行されたとホストシステム1000へ応答する。注意すべきは、メモリー管理ユニット204が第１論理ユニット360にマップされた実体ブロックを実際に消去する訳ではないので、フラッシュメモリー記憶装置100がクリアー指令を実行するための速度は、一般の書き換え可能DVD装置が同じ事を行う速度より格段に速いことである。

【００６０】

また、CPU 1102がフォーマット指令（例えば、書き換え可能DVD 指令の0×04指令）またはセクション閉鎖指令（例えば、書き換え可能DVD 指令の0×5B指令）をフラッシュメモリー記憶装置100へ送信する時、メモリー管理ユニット204が書き換え可能DVDブランクフラッグを「非ブランク状態」として標示する。

【００６１】

とりわけ、CPU 1102が読み出し指令（例えば、書き換え可能DVD 指令の0×28指令）をフラッシュメモリー記憶装置100へ送信して、第１論理ユニット360からデータを読み出す時、メモリー管理ユニット204は、書き換え可能DVDブランクフラッグが「ブランク状態」であるか否か判定する。もし書き換え可能DVDブランクフラッグが「ブランク状態」を標示している場合、メモリー管理ユニット204は、ヌル値（null value）をホストシステム1000へ直接送信する。もしも書き換え可能DVDブランクフラッグが「非ブランク状態」を標示している場合、メモリー管理ユニット204は、ホストシステム1000のために、読み出し指令により標示された論理アドレスに従って第１論理ユニット360にマップされた実体ブロックからデータを読み出す。

【００６２】

例えば、この実施形態中、メモリー管理ユニット204が書き換え可能DVDブランクフラッグをシステム領域302（図２右上）の実体ブロックへ保存する。しかし、この発明は、それに限定されるものではないとともに、この発明の他の実施形態中、書き換え可能DVDブランクフラッグは、また、他の非揮発メモリー（例えば、フラッシュメモリー制御器104中に追加的に配置された非揮発メモリー）へ保存することもできる。

【００６３】

この発明の実施形態中、メモリー管理ユニット204が書き込み容量パラメーターを確立する。ここで、書き込み容量パラメーターは、書き換え可能DVD装置の管理情報の１つであり、第１論理ユニット360の書き込み容量を記録する。詳細には、書き換え可能DVD装置中、データが書き換え可能DVDの蓄積アドレスへ順番に焼き付けられる。従って、ホストシステム1000は、第１論理ユニット360中の次の書き込み可能照会指令（例えば、書き換え可能DVD 指令の0×52指令）を使用して、第１論理ユニット360中の次の書き込み可能アドレスについて照会する。CPU 1102が書き込み指令（例えば、書き換え可能DVD 指令の0×2A指令）をフラッシュメモリー記憶装置100へ送信して、第１論理ユニット360へデータを焼き付ける時、メモリー管理ユニット204が書き込むべき論理アドレスに対応する論理ブロックにマップされた実体ブロックへデータを書き込むとともに、書き込みデータの容量に従って書き込み容量パラメーターを更新する。また、CPU 1102がフォーマット指令またはクリアー指令をフラッシュメモリー記憶装置100へ送信する時、メモリー管理ユニット204が書き込み容量パラメーターを初期化する（例えば、書き込み容量パラメーターがゼロにリセットされる）。とりわけ、CPU 1102が次の書き込み可能アドレス照会指令をフラッシュメモリー記憶装置100へ送信する時、メモリー管理ユニット204は、書き込み容量パラメーターに従って次の書き込み可能アドレスをホストシステム1000へ応答する。

【００６４】

特記すべきは、一般の書き換え可能DVD装置が書き換え可能DVD上でセクション閉鎖指令を実行する時、書き換え可能DVDの蓄積アドレスにおいてセクション閉鎖が発生するとともに、セクション閉鎖後に後続するデータが焼き付けられなければならないことである。従って、この実施形態中、CPU 1102がセクション閉鎖指令をフラッシュメモリー記憶装置100へ送信する時、たとえホストシステム1000が何のデータも書き込まなくても、メモリー管理ユニット204は、なお閉鎖セクションのサイズに従って書き込み容量パラメーターを更新する。

【００６５】

同様に、この実施形態中、メモリー管理ユニット204が書き込み容量パラメーターをシステム領域302（図２右上）の実体ブロックへ保存する。しかし、この発明は、それに限定されるものではないとともに、この発明の他の実施形態中、書き込み容量パラメーターは、また、他の非揮発メモリー（例えば、フラッシュメモリー制御器104中に追加的に配置された非揮発メモリー）へ保存することもできる。

【００６６】

この発明の他の実施形態中、ホストシステム1000のCPU 1102が読み出し指令または書き込み指令をフラッシュメモリー記憶装置100へ送信して、第１論理ユニット360からデータを読み出す又は第１論理ユニット360へデータを焼き付ける時、メモリー管理ユニット204が変換パラメーターを使用して、読み出し指令または書き込み指令により標示された論理アドレスおよびデータ容量を計算する。詳しくは、論理アクセスユニットがDVDシステム中に2048バイトを有する一方、論理アクセスユニットがフラッシュメモリーシステム中に512バイトを有する。従って、変換パラメーターが4（=2048/512）に設定される。例えば、CPU 1102が論理アドレス「512」から開始して512ユニットのデータを読み出そうとする時、メモリー管理ユニット204は、論理アドレス「2048」にマップされた実体アドレスから開始して2048ユニットのデータを読み出す。

【００６７】

この発明の実施形態中、ホストシステム1000のCPU 1102がエラー訂正データ照会指令をフラッシュメモリー記憶装置100へ送信する時、メモリー管理ユニット204は、不支持メッセージ（non-support message）をホストシステム1000へ送り返す。詳細には、DVDシステムが適切なエラー訂正アルゴリズムを使用してDVDに保存されたデータの正確性を確保する。この実施形態中、DVDシステムは、フラッシュメモリーシステムに適していない。かくして、エラー訂正ユニット256（図１Ｂ）がフラッシュメモリーシステムには適したエラー検査訂正（Error Checking and Correcting = ECC）コードを使用してデータ正確性を確保する。従って、CPU 1102が第１論理ユニット360からエラー訂正データを読み出そうとする時、ホストシステム1000からのエラー訂正データ照会指令を拒絶する。

【００６８】

DVDシステム中、ホストシステム1000は、分／秒／フレーム（Minute/Second/Frame = MSF）読み出し指令（例えば、書き換え可能DVD 指令の0×B9指令）を使用して、分／秒／フレームのパラメーターでデータを読み出すことができる。この発明の実施形態中、ホストシステム1000のCPU 1102がMSF読み出し指令をフラッシュメモリー記憶装置100へ送信する時、メモリー管理ユニット204は、下記の数式（１）を介してMSF読み出し指令により標示された論理アドレスを計算する。

$$LBA = ((M \times 60 + S - 2) \times 75 + F) \times 4 \cdots (1)$$

【００６９】

式中、LBAが前記フラッシュメモリー中のMSF読み出し指令に対応する論理アドレスを表し、MがMSF読み出し指令の分パラメーターを表し、SがMSF読み出し指令の秒パラメーターを表し、FがMSF読み出し指令のフレームパラメーターを表すものである。つまり、１分のデータが18000（= 60×75×4）の実体ブロック中に保存されるとともに、各実体アドレスが512バイトを有し、かつ１フレームのデータが４つの実体アドレス中に保存されるとともに、各実体アドレスが512バイトを有する。

【００７０】

DVDシステム中、ホストシステム1000がディスク排出（eject）指令を使用して書き換え可能なDVD装置中の書き換え可能なDVDを排出する。とりわけ、DVDシステム中、書き換え可能なDVDの焼き付けが完了した後、DVDの蓄積情報が変化しているので、ホストシステム1000がDVDを排出するよう要請するとともに、再びDVDが書き換え可能なDVD装置に挿入された後、DVDの蓄積情報を再読み出しする。しかし、フラッシュメモリー記憶システム中、全ての蓄積情報（例えば、論理ブロック-実体ブロックマップテーブル）が一時的にバッファーメモリー252（図１Ｂ）中に保存される。これにより、メモリー管理ユニット204は、フラッシュメモリー記憶装置100に対してプラグを抜いたり差し込んだりする（unplug and re-plug）必要なしに、第１論理ユニット360の最新情報を入手できる。この発明の実施形態中、ホストシステム1000のCPU 1102がディスク排出指令をフラッシュメモリー記憶装置100へ送信する時、メモリー管理ユニット204は、受信済みメッセージでホストシステム1000に応答するが、いかなる動作も起こさない。続いてCPU 1102が第１論理ユニット360を照会する時、メモリー管理ユニット204は、準備完了（ready）メッセージをホストシステム1000へ直接返送する。

【００７１】

この発明の実施形態中、メモリー管理ユニット204は、第１論理ユニット360が「書き込み保護状態」または「非書き込み保護状態」にあるかどうかを記録するために、書き込み保護フラッグを設定する。具体的には、ユーザーが応用プログラムを介してベンダー（vendor　製造業者）指令を使用することにより第１論理ユニット360の非書き込み保護フラッグを設定できる。例えば、もしもユーザーがデータを第１論理ユニット360に保存したくなくて誤って削除した場合、ユーザーは、ホストシステム1000にインストールされた応用プログラムを介してベンダー指令を発行することにより、メモリー管理ユニット204に「書き込み保護状態」として書き込み保護フラッグを設定するよう要請できる。

【００７２】

この発明の別の実施形態中、書き込み保護フラッグを変更する時、応用プログラムがユーザーに認証パスワードを尋ねるとともに、認証パスワードおよびセットされるデータをメモリー管理ユニット204へ送信する。そして、メモリー管理ユニット204がユーザーにより入力された認証パスワードをベンダーパスワードと比較する。メモリー管理ユニット204は、ユーザーにより入力された認証パスワードがベンダーパスワードと一致する時のみ、ユーザーの要求に従って書き込み保護フラッグを変更する。ここで、メモリー管理ユニット204は、書き込み保護フラッグおよびベンダーパスワードをシステム領域302（図２右上）に保存する。しかし、この発明は、それに限定されるものではないとともに、この発明の他の実施形態中、書き込み保護フラッグおよびベンダーパスワードは、また、他の非揮発メモリー（例えば、フラッシュメモリー制御器104中に追加的に配置された非揮発メモリー）へ保存することもできる。

【００７３】

この発明の実施形態中、ホストシステム1000のCPU 1102がクリアー指令、フォーマット指令またはセクション閉鎖指令をフラッシュメモリー記憶装置100に送信する時、メモリー管理ユニット204は、書き込み保護フラッグが「書き込み保護状態」であるかどうか判定する。書き込み保護フラッグが「書き込み保護状態」を標示している時、メモリー管理ユニット204は、エラーメッセージをホストシステム1000へ送り返すか如何なる動作も起こさない。エラーメッセージは、「書き込み保護されている（即ち、ASCコード0×27およびSenseKeyコード0×07）」、「媒体が存在しない（即ち、ASCコード0×3AおよびSenseKeyコード0×02）」、「媒体が変更中（即ち、ASCコード0×28およびSenseKeyコード0×06）」、「指令が支持されてない（即ち、ASCコード0×20およびSenseKeyコード0×05）」、「指令パラメーターエラー（即ち、ASCコード0×24およびSenseKeyコード0×05）」、「データ書き込みエラー（即ち、ASCコード0×03およびSenseKeyコード0×03）」または「データ書き込みエラー（即ち、ASCコード0×11およびSenseKeyコード0×03）」で有り得る。

【００７４】

もし書き込み保護フラッグが「書き込み保護状態」を標示していれば、上記したエラーメッセージの１つでホストシステム1000へ応答する以外に、この発明の別の実施形態中、メモリー管理ユニット204は、また、蓄積媒体特徴を直接に変更してホストシステム1000がいかなる書き込み又はクリアー指令を発行することを防止できる。例えば、ホストシステム1000のCPU 1102が蓄積媒体特徴照会指令をフラッシュメモリー記憶装置100に送る時、メモリー管理ユニット204は、書き込み保護フラッグが「書き込み保護状態」を標示しているかどうか判定する。もし書き込み保護フラッグが「書き込み保護状態」を標示していれば、メモリー管理ユニット204は、単一層読み出し専用DVDメッセージまたは二重層読み出し専用DVDメッセージをホストシステム1000へ返送して、第１論理ユニット360が読み出し専用DVDであることをホストシステム1000に通知する。

【００７５】

図４は、この発明の実施形態にかかる書き換え可能なデジタル多用途ディスク（ＤＶＤ）をシミュレーションする方法を示すフローチャート（流れ図）である。

【００７６】

図４において、ステップS401中、メモリー管理ユニット204が多数の論理ブロック（例えば、LBA(0)-LBA(L)）を配列するとともに、論理ブロックをフラッシュメモリーチップ106の実体ブロックの一部にマップする（図２）。

【００７７】

そして、ステップS403中、メモリー管理ユニット204が論理ブロックの一部を１つの論理ユニット（例えば、第１論理ユニット360）に分類する（図３）。ステップS405中、メモリー管理ユニット204が論理ユニットを書き換え可能なDVD蓄積媒体として配列する。

【００７８】

次に、ステップS407中、メモリー管理ユニット204は、論理ユニットが書き換え可能なDVD装置であるとホストシステム1000に宣言する。ステップS409中、メモリー管理ユニット204がホストシステム1000から書き換え可能なDVD指令を受信したことを識別するとともに、書き換え可能なDVD指令に従ってホストシステム1000から受信したデータを論理ブロックにマップされた実体ブロック中へ記録する。メモリー管理ユニット204が書き換え可能なDVD指令を識別するメカニズムは、上述したから、ここでは記述しない。

【００７９】

図５は、この発明の第２の実施形態にかかるフラッシュメモリー記憶システムを示すブロック図である。

【００８０】

図５において、フラッシュメモリー記憶システム500がカードリーダー502と取外し可能なメモリーカード504とを含む。

【００８１】

カードリーダー502がコネクター512とカードリーダー制御器514とメモリーカード連接インターフェースユニット516とを含む。

【００８２】

コネクター512がホストシステム1000（図１Ｂ）に連結するよう配列される。この実施形態中、コネクター512がUSBコネクターである。しかし、この発明は、それに限定されるものではないとともに、コネクター512は、また、PATAコネクター、SATAコネクター、IEEE 1394コネクター、PCI高速コネクター、IEDコネクターまたは他の適合する伝送コネクターで有り得る。この実施形態中、カードリーダー502がコネクター512を介して分離方式で（in a detachable manner）ホストシステム1000に連結される。しかし、この発明は、それに限定されるものではないとともに、この発明の他の実施形態中、カードリーダー502は、固定方式で（in fixed manner）ホストシステム1000に連結され得る。

【００８３】

カードリーダー制御器514がコネクター512に連結されるとともに、カードリーダー502の操作を制御するよう配列される。とりわけ、この実施形態中、カードリーダー制御器514は、カードリーダー502が書き換え可能なDVD装置であるとホストシステム1000に対して

宣言する。例えば、カードリーダー制御器514は、カードリーダー502がDVD+RW規格、DVD-RW規格、DVD-RAM規格またはBD規格に適合するディスク装置であるとホストシステム1000に対して宣言する。また、カードリーダー制御器514は、ホストシステム1000から受信した書き換え可能なDVD指令を識別し、かつ書き換え可能なDVD指令に従って取外し可能なメモリーカード504を操作する。ホストシステム1000から受信した書き換え可能なDVD指令に応答するメカニズムは、上述したので、ここでは記述しない。

【００８４】

メモリーカード連接インターフェースユニット516がカードリーダー制御器514に連結されるとともに、取外し可能なメモリーカード504に連結するよう配列される。

【００８５】

取外し可能なメモリーカード504が分離可能な方式でカードリーダー502にプラグされる。取外し可能なメモリーカード50がメモリーカード制御器522とフラッシュメモリーチップ524とを含む。

【００８６】

メモリーカード制御器522が論理ブロックを配列するとともに、論理ブロックを実体ブロックにマップするので、ホストシステム1000がカードリーダー502を介して取外し可能なメモリーカード504中のフラッシュメモリーチップ524にアクセスできる。

【００８７】

具体的には、カードリーダー制御器514は、取外し可能なメモリーカード504が書き換え可能なDVDであるとホストシステム1000に対して宣言するとともに、ホストシステム1000からの書き換え可能なDVD指令を識別し、かつメモリーカード制御器522がフラッシュメモリーチップ524のブロック管理メカニズムを実行し、かつカードリーダー制御器514の指令に従ってフラッシュメモリーチップ524を操作する。

【００８８】

この実施形態中、フラッシュメモリーチップ524がフラッシュメモリーチップ106と同一であるから、フラッシュメモリーチップ524の構造および機能は、ここでは記述しない。

【００８９】

以上のごとく、この発明を実施形態により開示したが、もとより、この発明を限定するためのものではなく、当業者であれば容易に理解できるように、この発明の技術思想の範囲内において、適当な変更ならびに修正が当然なされうるものであるから、その特許権保護の範囲は、特許請求の範囲および、それと均等な領域を基準として定めなければならない。

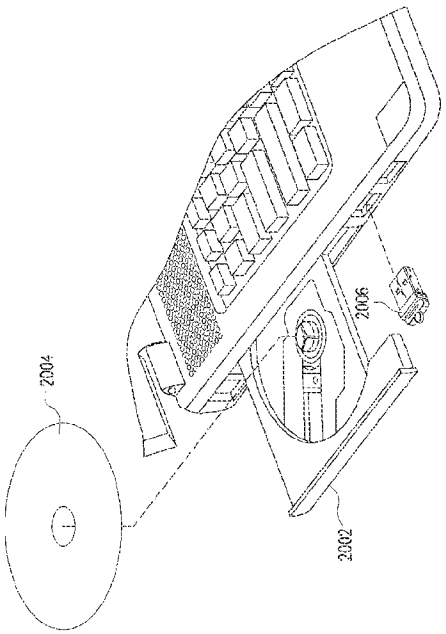
【符号の説明】

【００９０】

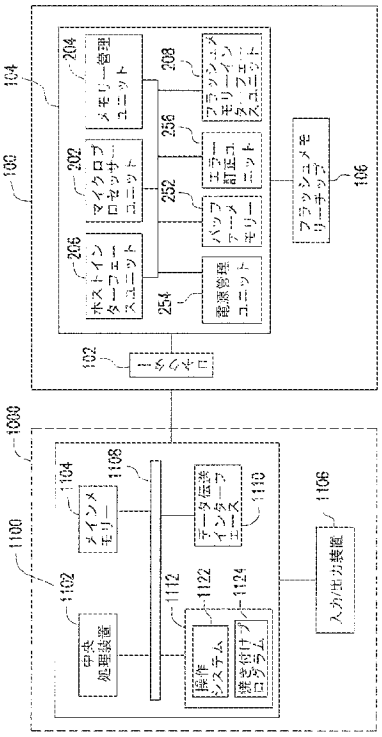
100 フラッシュメモリー記憶装置
102 コネクター
104 フラッシュメモリー制御器
106 フラッシュメモリーチップ
202 マイクロプロセッサーユニット
204 メモリー管理ユニット
206 ホストインターフェースユニット
208 フラッシュメモリーインターフェースユニット
252 バッファーメモリー
254 電源管理ユニット
256 エラー訂正ユニット
302 システム領域
304 蓄積領域
304a データ領域
304b 予備領域
306 交替領域

360　第１論理ユニット
380　第２論理ユニット
502　カードリーダー
504　取外し可能なメモリーカード
512　コネクター
514　カードリーダー制御器
516　メモリーカード連接インターフェースユニット
522　メモリーカード制御器
524　フラッシュメモリーチップ
1000　ホストシステム
1100　コンピューター
1102　中央処理ユニット
1104　メインメモリー
1106　入力／出力装置
1108　システムバス
1110　データ伝送インターフェース
1112　メインディスクドライブ
1122　操作システム（Operation System = OS）
1124　焼き付けプログラム
1202　マウス
1204　キーボード
1206　ディスプレイ
1208　プリンター
1212　フラッシュドライブ1214　メモリーカード
1216　固態ドライバー（Solid State Driver = SSD）
1218　嵌め込み式記憶装置
2002　デジタル多用途ディスク（Digital Versatile Disc = DVD）ドライブ
2004　デジタル多用途ディスク（Digital Versatile Disc = DVD）
2006　フラッシュドライブ
PBA(O)-PBA（N）　実体ブロック
LBA(O)-LBA(L)　論理ブロック

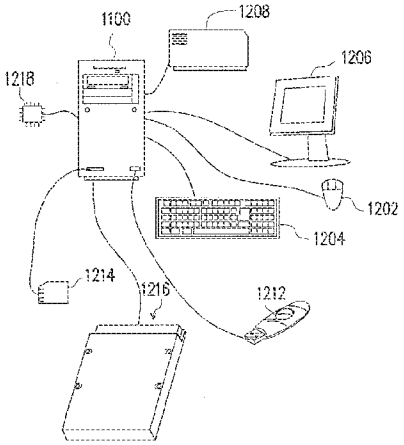
【図１Ａ】



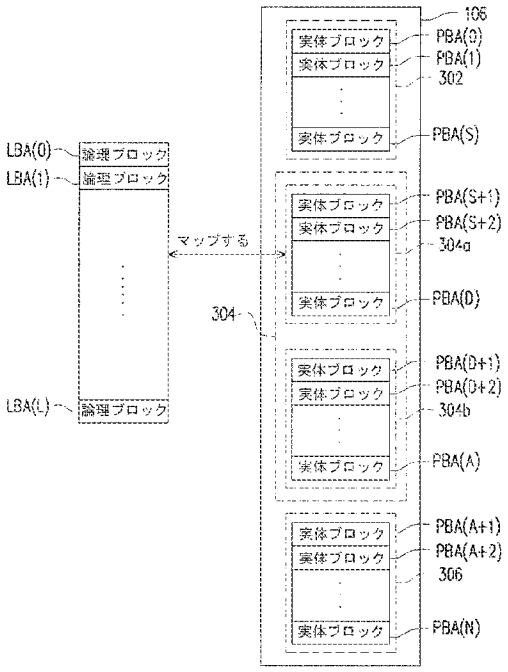
【図１Ｂ】



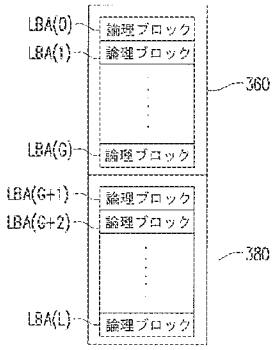
【図１Ｃ】



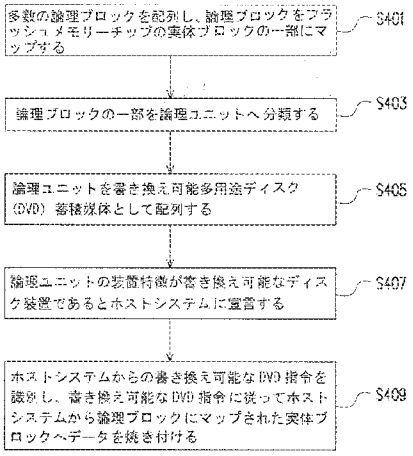
【図２】



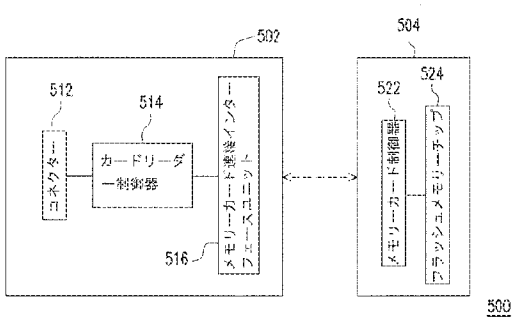
【図３】


LBA(0)
論理ブロック
LBA(1)
論理ブロック
360
LBA(G)
論理ブロック
LBA(G+1)
論理ブロック
LBA(G+2)
論理ブロック
380
LBA(L)
論理ブロック

【図４】


多数の論理ブロックを配列し、論理ブロックをフラッシュメモリーチップの実体ブロックの一部にマップする
S401
論理ブロックの一部を論理ユニットへ分類する
S403
論理ユニットを書き換え可能多用途ディスク(DVD)蓄積媒体として配列する
S405
論理ユニットの装置特徴が書き換え可能なディスク装置であるとホストシステムに宣言する
S407
ホストシステムからの書き換え可能なDVD指令を識別し、書き換え可能なDVD指令に従ってホストシステムから論理ブロックにマップされた実体ブロックへデータを焼き付ける
S409

【図５】


502
504
512
514
524
522
コネクター
カードリーダー制御器
メモリーカード媒体インターフェースユニット
516
メモリーカード制御器
フラッシュメモリーチップ
500

フロントページの続き

(51)Int.Cl.		F I		テーマコード（参考）
G06K 17/00	*(2006.01)*	G06F 3/08	H	
		G06K 17/00	L	

(72)発明者 許 世賢
台湾彰化縣芬園鄉中嵙村大彰路二段２２１巷七號
(72)発明者 游 祥雄
台湾新竹縣▲弓▼林鄉光復街８９巷１２號

Ｆターム(参考) 5B035 AA06 BB09 CA29
5B058 CA23 CA26
5B060 AB26
5B065 BA01 BA09 CC02 ZA01
5B082 FA05 JA06

【外国語明細書】
2011123863000001.pdf

JP2011203916A MEMORY CONTROLLER AND SEMICONDUCTOR STORAGE DEVICE

Data originating from sources other than the EPO may not be accurate, complete, or up to date.

The wording below is an initial machine translation of the original publication. To generate a version using the latest translation technology, go to the original language text and use Patent Translate.

Memory controller and semiconductor memory device

[0001] The present invention relates to a memory controller and a semiconductor memory device.

[0002] The NAND flash memory simplifies the structure by collectively erasing data stored in a plurality of memory cells called blocks, and realizes a low cost and a large capacity. Furthermore, NAND flash memories are widely used as storage devices in place of HDDs, host storage devices such as mobile phones or portable music players because they have no moving parts and low power consumption. However, it is known that the NAND flash memory has a limit on the number of times of writing / erasing.

[0003] 特開２００４－３２６５２３号公報

[0004] The present invention provides a memory controller and a semiconductor memory device capable of averaging the number of block erases with high accuracy.

[0005] According to one aspect of the present invention, there is provided a memory controller that performs control for rewriting data in a nonvolatile semiconductor memory having a plurality of blocks that are units of data erasing, wherein the logical address for each block specified by a host is A first management table for storing a correspondence between a logical block address of the block and a physical block address indicating a storage position of the block in the nonvolatile semiconductor memory, and the number of times data is written from the host to the nonvolatile semiconductor memory In response to a write request designating a logical address from the host, a second management table that stores the logical erase address for each logical block address, a third management table that stores the number of data erases for each physical block address, and Spare block that is the block that does not have a corresponding logical address A write control unit that selects and writes data to the spare block, and based on the number of data writes stored for each logical block address and the number of data erases stored for each physical block address A memory controller is provided that averages among the plurality of blocks.

[0006] According to the present invention, it is possible to provide a memory controller and a semiconductor memory device capable of averaging the number of block erases with high accuracy.

[0007] FIG. 1 is a block diagram showing a configuration of a memory system according to the first to third embodiments. FIG. 2 is a diagram showing the relationship between physical addresses and logical addresses in the semiconductor memory devices of the first to third embodiments. FIG. 3 illustrates a logical address management table according to the first and third embodiments. FIG. 4 is a diagram illustrating a spare block management table according to the first and third embodiments. FIG. 5 is a flowchart illustrating a specific procedure of passive wear leveling performed by the memory controller according to the first embodiment. FIG. 6 illustrates a logical address management table according to the second embodiment. FIG. 7 is a diagram showing a distribution of the number of times of writing for each logical address. FIG. 8 is a diagram in which FIG. 7 is grouped according to the number of times of writing. FIG. 9 illustrates a spare block management table according to the second embodiment. FIG. 10 is a diagram showing the distribution of the number of block erases for each spare block. FIG. 11 is a diagram in which FIG. 10 is grouped according to the magnitude relationship of the number of block erases. FIG. 12 is a flowchart illustrating a specific procedure of passive wear leveling performed by the memory controller according to the second embodiment. FIG. 13 is a flowchart

illustrating a specific procedure of active wear leveling performed by the memory controller according to the third embodiment.

[0008] The limitation on the number of times of writing / erasing to the memory cell of the NAND flash memory is that a high voltage is applied to the gate with respect to the substrate and electrons are injected into the floating gate (writing), and a high voltage is applied to the source with respect to the gate. This is caused by the fact that electrons are extracted from the floating gate (erasing). That is, if the write / erase process is performed many times on a specific memory cell, the oxide film around the floating gate may deteriorate and the data may be destroyed.

[0009] In order to avoid such a write / erase process from being concentrated on a specific memory cell, the memory controller attempts to average the number of write / erase processes. This is because the memory controller counts the number of erasures of physical blocks, which are units of erasure processing, and replaces the physical blocks with a large number of erasure processing times with the physical blocks with a small number of erasing operations to average the number of write / erase processing times. It is implemented by leveling.

[0010] However, in conventional wear leveling, only the number of times of writing / erasing as an attribute of the physical block is referred to (for example, see Patent Document 1). In Patent Document 1, an attempt is made to average the number of times of writing by providing a block exchanging means for exchanging a physical block with the smallest number of times of writing and a physical block whose number of times of writing reaches a predetermined number of checks.

[0011] 2. Description of the Related Art A semiconductor memory device that performs wear leveling generally uses a logical address / physical address conversion (hereinafter referred to as "logical / physical conversion") table for converting a logical address into a physical address. By using this logical-physical conversion table, it is possible to distribute physical storage positions inside the semiconductor memory device even when there is a bias in the update position specified by the logical address by the host. In a semiconductor memory device that performs wear leveling, the logical / physical address correspondence changes each time rewriting is performed.

[0012] Unbalanced rewrite frequency as an external request from the host when the correspondence between logical address / physical address changes each time rewrite is performed, such as a solid state drive (SSD) or other wear leveling semiconductor memory device Is not an attribute of a physical block but an attribute of a logical address. However, in the conventional wear leveling, only the number of times of writing / erasing as the physical block attribute is referred to as described above, and the bias characteristic of the rewrite frequency as the logical address attribute is not considered at all.

[0013] Therefore, in conventional wear leveling, for example, a physical address of a physical block with a small number of write / erase times may be assigned to a logical address with a low rewrite frequency as an external request from the host. In this case, the physical block is unlikely to be updated in relation to the new logical-physical conversion after the allocation, and it is highly likely that the number of times of writing / erasing remains low. As a result, the number of times of rewriting the physical block is averaged. There was a problem that it was not done properly.

[0014] Exemplary embodiments of a memory controller and a semiconductor memory device will be explained below in detail with reference to the accompanying drawings. Note that the present invention is not limited to these embodiments.

[0015] (First Embodiment) FIG. 1 is a block diagram showing a configuration of a semiconductor memory device 2 as a memory system according to a first embodiment of the present invention. The semiconductor storage device 2 is a memory card that is detachably connected to a host 3 such as a personal computer or a digital camera, or an embedded type memory that is housed inside the host 3 and functions as an external storage device of the host 3. .

[0016] The semiconductor memory device 2 includes a nonvolatile semiconductor memory (hereinafter also simply referred to as "memory unit") 40 and a memory controller 10. The memory unit 40 is, for example, a NAND flash memory, and has a structure in which a large number of memory cells 41 as unit cells are arranged in a matrix at intersections of bit lines (not shown) and word lines 42. Data erasure in the memory unit 40 is performed in units of physical blocks including a plurality of unit cells, and the memory unit 40 includes a

plurality of physical blocks. Writing / reading to / from the memory unit 40 is performed in units of physical pages. Since one physical block is composed of a plurality of physical pages, the size of the physical page is smaller than the size of the physical block.

[0017] The memory controller 10 includes a CPU 20, which is a control unit, a RAM 30, a host I / F (interface) 12, a ROM 13, and error correction (ECC) that performs encoding processing of stored data and decoding processing of stored data. : Error Correcting Code) circuit 14 and a NAND I / F (interface) 15, which are connected via a bus 11.

[0018] The memory controller 10 performs data transmission / reception between the host 3 and the RAM 30 via the host I / F 12 and performs data transmission / reception between the memory unit 40 and the RAM 30 via the NAND I / F 15 under the control of the CPU 20. . In addition to the normal control unit that performs data transmission / reception control between the host 3 and the RAM 30 and data transmission / reception control between the memory unit 40 and the RAM 30, the CPU 20 determines the number of times data is written to the memory unit 40 for each logical address. The number-of-writes counting unit 21 that counts, the number-of-erasing-counting unit 22 that counts the number of data erasures as the number of data rewrites for each physical block that is a unit of data erasure in the memory unit 40, and the physical block to which the corresponding logical address is not assigned A spare block selection processing unit 23 for selecting a certain spare block is provided. A spare block (free block) is a physical block that does not contain valid data and is not allocated for use.

[0019] When a data write request is generated from the host 3, the spare block selection processing unit 23 accesses the RAM 30 via the bus 11, and based on management information (for example, a spare block management table described later) held therein. The spare block to which data is written is selected.

[0020] FIG. 2 shows the concept of logical-physical conversion in the present embodiment, and the premise terms will be described. The logical address is an address used by the host, for example, LBA (Logical Block Addressing). LBA is a logical address in which a serial number from 0 is assigned to a sector (size: eg, 512 B). A sector is a unit smaller than the physical page described above. The physical address is an address indicating a storage position in the memory unit 40.

[0021] In this embodiment, as a logical address unit having the same partition size as the physical block size that is a data erasing unit, a logical block address that is a higher address of the logical address (for example, LBA) is used. It is associated with the physical block address which is the physical address of. That is, as shown in FIG. 2, logical-physical conversion in which one physical block address is associated with one logical block address is performed. However, as long as it is within the scope of the technical idea included in the gist of the present embodiment, the logical-physical conversion relationship is not necessarily limited.

[0022] The RAM 30 includes a logical address management table 31 and a spare block management table 32 in addition to an area functioning as a cache area interposed between the host 3 and the memory unit 40. The logical address management table 31 includes a logical address / physical address conversion table (logical / physical conversion table) 70 which is a first management table for converting a logical block address into a physical block address in the memory unit 40 as shown in FIG. It is out.

[0023] As described above, since the spare block selection processing unit 23 selects a spare block to which a corresponding logical address is not assigned in response to a data write instruction from the host 3, in this embodiment, data rewriting is performed. Each time, the relationship between logical and physical conversion will change. A write instruction to the memory unit 40 from the host 3 usually includes a head logical address and a data size.

[0024] Further, as shown in FIG. 3, the logical address management table 31 is a host for each physical block data erase count (physical block erase count) indicated by the physical block address registered in the logical-physical conversion table 70 and for each logical block address. 3 stores the number of times of writing (number of times of logical address writing) designated by 3.

[0025] Accordingly, the logical address management table 31 also includes a logical write count table which is a second management table for storing the write count from the host 3 for each logical block address. The logical write count table reflects the rewrite frequency bias characteristic as an attribute of the logical address.

[0026] As shown in FIG. 4, the spare block management table 32 has a physical block address of a spare block, which is a physical block to which a corresponding logical address is not assigned, and a data erase count (physical block erase count) of the corresponding physical block. Is remembered. In this table, for example, as shown in FIG. 4, physical block addresses may be sorted according to the number of physical block erase times.

[0027] Therefore, the information on the number of times of data erasure of the physical block registered for each physical block address registered in the logical address management table 31 and the information on the number of times of data erasure of the physical block registered for each physical block address registered in the spare block management table 32 are obtained. In addition, a physical erase count table, which is a third management table for storing the data erase count for each physical block address, is configured. The physical erase count table reflects the frequency of erase (and write) for each physical block, that is, the physical fatigue level of the physical block.

[0028] The CPU 20 executes an operation as an address conversion unit (not shown) that performs conversion between a logical address and a physical address based on the logical-physical conversion table 70 included in the logical address management table 31 by firmware (FW). Control of the entire semiconductor memory device 2 in response to a command input from the host 3 is also executed in the CPU 20 by firmware.

[0029] The ROM 13 is a storage unit in which a boot program or the like of the semiconductor storage device 2 is stored, and the memory controller 10 controls the semiconductor storage device 2 in a part of the semiconductor memory unit 30 or a non-illustrated nonvolatile storage unit. Information is stored.

[0030] In the present embodiment, the CPU 20 included in the memory controller 10 has a logical address that the write count counter 21 counts for each logical block address when the host 3 receives an instruction to write data to an arbitrary logical address. Increment the number of writes. The result is stored in the logical address write count column of the logical address management table 31 in the RAM 30 via the bus 11.

[0031] Further, the CPU 20 increments the physical block erase count counted by the erase count counting unit 22 for each physical block address even when data is erased in each physical block of the memory unit 40. As a result, if the physical block has a corresponding logical address (that is, logical block address), the physical block address of the physical block in the logical address management table 31 in the RAM 30 is registered via the bus 11. Stored in the column of the physical block erase count in the row (that is, the row in which the logical block address is registered).

[0032] If the physical block from which data has been erased is a spare block to which a corresponding logical address is not assigned, the physical block address of the physical block in the spare block management table 32 in the RAM 30 is stored via the bus 11. It is stored in the physical block erase count column of the stored row.

[0033] Conventionally, wear leveling (hereinafter referred to as passive wear leveling) performed at the time of a write request from the host 3 is the number of past write (rewrite) requests to the logical address designated by the write request from the host 3. It was done without any consideration. That is, in the conventional passive wear leveling, for example, depending on only the number of erasures of the spare block, for example, a spare block with a small number of physical block erasures is selected in the spare block management table 32 of FIG. Block erasing and writing) and assigning a logical address designated by the host to the physical block.

[0034] However, in such passive wear leveling, there is a possibility that the purpose of wear leveling, that is, averaging the number of erasures, cannot be realized properly. For example, when a rewrite request happens to occur at a logical address that is less frequently rewritten as an external request from the host, there is a possibility that data is written by selecting a physical block with a small number of erases from the spare block. In this case, there is a high possibility that the physical block has a small number of erasures even in relation to the new logical-physical conversion after the logical address assignment, resulting in a bias in the number of physical block rewrites and averaging. There was a problem that it was not done properly.

[0035] Therefore, in the present embodiment, when the write frequency of a logical address for which data is written (rewritten) from the host is low, a spare block with a large number of erases is selected for the logical address. The logical address is assigned and the data is written to the selected spare block. On the other hand, if the write frequency of the logical address that is instructed by the host is high, the spare address with a small number of erases is selected and assigned to the logical address, and the logical address is assigned to the selected spare block. Write data. Thereby, it is possible to solve the problem that the above-described averaging is not appropriately performed.

[0036] Specifically, the spare block selection processing unit 23 (write control unit) responds to a write request specifying a logical block address with a relatively large number of writes stored in the logical address management table 31 with a spare block. A spare block with a relatively small number of erases stored in the management table 32 is selected, and a spare block with a relatively large number of erases is selected in response to a write request specifying a logical address with a relatively small number of writes. select.

[0037] For example, when a write request specifying the logical block address "n−1" in FIG. 3 is received from the host 3, it is determined that the logical address write count "150" in the logical address management table 31 is relatively small. Thus, the spare block of the physical block address "WWW" in FIG. 4 having a relatively large number of physical block erases in the spare block management table 32 is selected and data is written. Conversely, when a write request specifying the logical block address "n" in FIG. 3 is received, it is determined that the logical address write count "400" in the logical address management table 31 is relatively large, and the spare block The spare block with the physical block address "aaa" in FIG. 4 having a relatively small number of physical block erases in the management table 32 is selected and data is written.

[0038] More specifically, the logical block addresses sorted in the logical address write count size relationship in the logical address management table 31 in FIG. 3 and the spare block sorted in the physical block erase count size in the spare block management table 32 in FIG. The physical block addresses of the blocks may be associated with each other so that the magnitude relationship between them is reversed.

[0039] Further, as a method of associating the magnitude relationship in reverse order, it is possible to use a method that does not use the raw count value as described above. For example, a "write number relative value" column is further provided in the logical address management table 31 of FIG. 3, and a value (0 to 1) obtained by dividing each logical address write number by the maximum value of the logical address write number at that time. Stored for each logical block address. The spare block management table 32 is also provided with a column "relative value of erase count", and a value obtained by dividing each physical block erase count by the maximum value of the physical block erase count in the spare block at that time (0 to 0). 1) is held for each spare block.

[0040] Then, when the "write count relative value" of the logical block address in FIG. For example, according to the following (Equation 1), a "relative value of the number of preferable erasures" Y is calculated.

[0041] Y = 1-X (Formula 1)

[0042] The spare block selection processing unit 23 selects the spare block having the "erasing number relative value" closest to Y obtained here from the spare block management table 32 and writes the data. For example, for a write request specifying a logical block address with a relatively small number of writes, such as "relative number of write times" X = 0.2, the "relative value of preferred erasure number" Y = 0.8 is the most. A spare block having a relatively large number of erasures is selected from the spare blocks.

[0043] The above (formula 1) is not limited to the above formula as long as X and Y are in a reverse relationship. Further, the correspondence is made more uniform by using a non-linear expression that reflects the deviation of the write frequency for each logical address from the host shown in FIG. 7 and the deviation of the block erase frequency for each spare block shown in FIG. You may devise as follows.

[0044] A specific procedure of passive wear leveling performed by the memory controller 10 of the present embodiment will be described below using the flowchart of FIG.

[0045] ⟨Step S101⟩ Data Write Instruction A data write request specifying the logical block address LA1 is received from the host 3 to the memory controller 10, for example.

[0046] ⟨Step S102⟩ Selection of Spare Block The spare block selection processing unit 23 selects the spare block SB1 for the designated logical block address LA1 based on the above (Equation 1).

[0047] ⟨Step S103⟩ Block Erasing and Data Writing Block erasing the physical block at the physical block address in the memory unit 40 corresponding to the spare block SB1 selected in step S102, and transferring the data instructed to the physical block from the host 3 Write.

[0048] ⟨Step S104⟩ Incrementing the Physical Block Erase Count The erase count counter 22 in the CPU 20 counts the block erase in step S103 for each physical block address, and the physical block address in the spare block management table 32 of FIG. Increase the physical block erase count by 1.

[0049] ⟨Step S105⟩ Increment of Logical Address Write Count The write count counter 21 in the CPU 20 counts the write instruction to each logical block address from the host 3 in step S101 for each logical block address. The logical address write count of the logical block address LA1 in the address management table 31 is increased by one.

[0050] ⟨Step S106⟩ Replacement of Physical Block Further, physical block address and physical block erase count described in spare block management table 32 of spare block SB1 selected in step S102, and physical block of logical block address LA1 in logical address management table 31 Exchange address and physical block erase count. At the time of replacement, for example, a predetermined storage area (not shown) in the RAM 30 may be used as a temporary storage area. As a result, the physical block entered in the logical address management table 31 and the physical block entered in the spare block management table 32 are replaced. After the replacement, the spare block management table 32 is sorted according to the size of the physical block erase count (data erase count).

[0051] ⟨Step S107⟩ The table sort spare block management table 32 is sorted according to the magnitude relation of the physical block erase count (data erase count).

[0052] Steps S101 to S107 are repeated each time an instruction to write data to an arbitrary logical address is received from the host 3 to the memory controller 10.

[0053] As described above, in the present embodiment, when the write frequency of the logical address for which data is instructed from the host is low, the spare block with a large erase count is selected and the data is written and the host writes When the frequency of writing the designated logical address is high, a spare block with a small number of erasures is selected and the data is written.

[0054] As described above, the semiconductor memory device 2 according to the present embodiment refers to the number of rewrites, which is the attribute of the logical address, that is, the rewrite frequency of the logical address when the relationship of logical-physical conversion changes every time data is rewritten. The number of physical block rewrites is averaged.

[0055] In this embodiment, in passive wear leveling, a physical (spare) block having a large number of erases and a large degree of physical fatigue is assigned to a logical block address having a low write frequency, and conversely, a logical block having a high write frequency. By allocating physical (spare) blocks with a small degree of physical fatigue with a small number of erasures to addresses, it is considered possible to improve the accuracy of averaging the number of rewrites of physical blocks.

[0056] That is, the phenomenon of selecting a physical block with a small number of write / erase times as a spare block at a logical address with a low rewrite frequency, which has occurred in the conventional passive wear leveling, can be avoided. It is possible to achieve the purpose of wear leveling, which is a higher level of accuracy, and to extend the life of the memory system compared to the conventional one.

[0057] On the other hand, even when there is no data rewrite instruction from the host, the memory system voluntarily transfers the data written in the physical block with a small number of erases to the spare block with a large number of erases. There is a method of wear leveling (hereinafter referred to as active wear leveling) in which a physical block with a small number is a spare block.

[0058] In the above-described embodiment, it is considered that the required number of active wear leveling is reduced by increasing the accuracy of the number of times of physical block erase for passive wear leveling. Therefore, it is possible to reduce the number of physical block erasures that accompany voluntary data rewriting of the memory system in active wear leveling, and this can be expected to increase the life of the memory system.

[0059] (Second Embodiment) A block diagram showing a configuration of a semiconductor memory device 2 as a memory system according to a second embodiment of the present invention is the same as FIG. Description of the same components is omitted. Also in the present embodiment, as shown in FIG. 2 as an example, logical-physical conversion in which one physical block address, that is, one physical block of the memory unit 40 is

associated with one logical block address is performed. To be done. However, the logical-physical conversion relationship is not limited as long as it does not depart from the gist of the present embodiment.

[0060] Further, in response to a data write instruction from the host 3 to an arbitrary logical address, a spare block (with no logical address assigned) is selected, data is written, and associated with the logical address. That is, the place where the relation of logical-physical conversion changes each time data is rewritten is the same as in the first embodiment.

[0061] In the present embodiment, the processing of the spare block selection processing unit 23 and the configurations of the logical address management table 31 and the spare block management table 32 provided in the RAM 30 are different from those of the first embodiment.

[0062] As in the first embodiment, the logical address management table 31 in this embodiment shown in FIG. 6 is a logical block that converts a logical block address into a physical block address that is a physical address in the memory unit 40 of the corresponding physical block. An address / physical address conversion table (logical / physical conversion table) 70 is included. Stores the data erase count (physical block erase count) of the physical block address described in the logical-physical conversion table 70, stores the write count (logical address write count) for each logical block address, and includes a logical write count table. It is also the same as in the first embodiment.

[0063] However, as shown in FIG. 6, the logical address management table 31 of this embodiment further stores a logical address group index for each logical block address. In general, there is a bias in the frequency of writing from the host 3 for each logical address as shown in FIG. Therefore, it is possible to classify all logical block addresses into one of a plurality of logical address groups based on the magnitude relationship of the number of logical address writes in FIG. For example, as shown in FIG. 8, it can be classified into four logical address groups each having indexes A to D.

[0064] Specifically, for example, the ratio (0 to 1) of each logical block address write count to the maximum logical address write count for all the logical block addresses shown in FIG. , 0.5, and 0.75, and the four logical address groups having indexes A, B, C, and D from the one with the larger ratio as shown in FIG. Can be classified.

[0065] The number of logical address groups is not limited to four as long as it is plural. Also, based on the magnitude relationship of the logical address write count, all the logical block addresses may be classified into a plurality of logical address groups having approximately the same number of logical block addresses, and based on the magnitude relationship of the logical address write count. The classification is not limited to the above classification method including the threshold interval. The index determined in this way is stored for each logical block address as an attribute of each logical block address in the logical address group column of the logical address management table 31 of FIG.

[0066] On the other hand, the spare block management table 32 in the present embodiment shown in FIG. 9 also corresponds to the physical block address of a spare block that is a physical block to which the corresponding logical address is not assigned, as in the first embodiment. The physical block erase count (data erase count) of the physical block is stored. The physical block addresses are sorted according to the number of physical block erase times.

[0067] However, as shown in FIG. 9, the spare block management table 32 of this embodiment further stores a spare block group index for each physical block address of the spare block.

[0068] In general, there is a bias in the frequency of block erases for each spare block as shown in FIG. Accordingly, it is possible to classify all spare blocks into one of the number of logical block groups, for example, the same number of spare block groups based on the magnitude relationship of the physical block erase count in FIG. For example, as shown in FIG. 11, it can be classified into four spare block groups each having indexes A to D.

[0069] Specifically, for example, the ratio (0 to 1) of each physical block erase count to the maximum value of the physical block erase count for all the spare blocks shown in FIG. 9 is set to three threshold values, for example, 0.25, 0 .5 and 0.75 are classified into four, and from the smaller ratio as shown in FIG. 11, the blocks are classified into spare block groups having indexes A, B, C, and D, respectively. it can.

[0070] The number of spare block groups may be the same as the number of logical address groups, and is not limited to four. Also, based on the magnitude relationship of the physical block erase count, all spare blocks may be classified into a plurality of spare block groups having approximately the same number of spare blocks. If the classification is based on the physical block erase count magnitude relationship For example, the classification method including the threshold interval is not limited. This index is stored in the spare block group column of the spare block management table 32 of FIG. 9 for each physical block address of the spare block.

[0071] In the present embodiment, when a logical block address for which a data write (rewrite) instruction has been issued from the host belongs to a logical address group with a low write frequency, the frequency of erasures for that logical block address. A spare block belonging to a larger spare block group is selected, the logical block address is assigned, and the data is written to a physical block in the memory unit 40 corresponding to the selected spare block.

[0072] On the other hand, if the logical block address for which a write instruction has been issued from the host belongs to a logical address group with a high write frequency, a spare block belonging to a spare block group with a low frequency of erasure is assigned to the logical block address. Is selected, the logical block address is assigned, and the data is written to the physical block in the memory unit 40 corresponding to the selected spare block.

[0073] In the example of FIGS. 8 and 11 described above, each logical address group has a spare block group of the same index, that is, the relationship between the logical address write count and the physical block erase count is reversed. Combine A's, B's, C's, and D's. From this combination, a spare block is selected for a logical address for which a write instruction has been issued.

[0074] A specific procedure of passive wear leveling performed by the memory controller 10 of the present embodiment will be described below using the flowchart of FIG.

[0075] ⟨Step S201⟩ Data Write Instruction A data write request specifying the logical block address LA1, for example, comes from the host 3 to the memory controller 10.

[0076] ⟨Step S202⟩ Logical Address Group Determination The spare block selection processing unit 23 accesses the logical address management table 31 of FIG. 6 and determines a logical address group based on the logical address group index of the logical block address LA1.

[0077] ⟨Step S203⟩ Selection of Spare Block The spare block selection processing unit 23 accesses the spare block management table 32 of FIG. 9, and selects a spare block SB1 belonging to a spare block group having the same index as the logical address group determined in step S202. select.

[0078] As described above, a spare block group is associated with each logical address group so that the magnitude relationship between the number of logical address writes and the magnitude relationship between the physical block erase counts are reversed. Accordingly, any spare block belonging to the spare block group corresponding to the logical address group determined in step S202 is selected, for example, one spare block belonging to the spare block group is selected at random. It doesn't matter.

[0079] However, the same idea is applied according to the relative magnitude of the logical address write count of the logical block address LA1 in the logical address group determined in step S202, reflecting the idea of the first embodiment. If the spare block SB1 is selected from the index spare block group, the number of physical block erasures can be averaged more effectively. That is, when the logical address write count of the logical address LA1 is relatively large in the logical address group determined in step S202, the physical block erase count is relatively small among the spare blocks belonging to the corresponding spare block group. If a spare block is selected and the logical address write count is relatively large, a spare block with a relatively small physical block erase count may be selected.

[0080] Such a mechanism can be realized, for example, by using (Equation 1) described in the first embodiment. In this case, the logical address group column of the logical address management table 31 in FIG. 6 is added to the logical address group index, and the logical address write count of the row is the maximum write count in the logical address group of the index. Further, a "write number relative value" for each logical address group, which is a value obtained by dividing the number, is stored. For example, "A: 0.3". Similarly, in the spare block group column of the spare block management table 32 of FIG. 9, the value obtained by dividing the physical block erase count of the row by the maximum erase count within each spare block group is "Erase" for each

spare block group. The relative number of times "is additionally stored as" D: 0.4 ". Then, after selecting a spare block group having the same index as the logical address group determined in step S202, the spare block group SB1 closest to the "preferred erase count relative value" is selected according to (Equation 1) and the like.

[0081] ⟨Step S204⟩ Block Erasing and Data Writing Block erasing the physical block at the physical block address in the memory unit 40 corresponding to the spare block SB1 selected in step S203, and transferring the data instructed to the physical block from the host 3 Write.

[0082] ⟨Step S205⟩ Incrementing the Physical Block Erase Count The erase count counting unit 22 in the CPU 20 counts the block erase in step S204 for each physical block address, and the physical block address in the spare block management table 32 of FIG. Increase the physical block erase count by 1.

[0083] ⟨Step S206⟩ Increment of Logical Address Write Count The write count counter 21 in the CPU 20 counts the write instruction to each logical block address from the host 3 in step S201 for each logical block address. The logical address write count of the logical block address LA1 in the address management table 31 is increased by one.

[0084] ⟨Step S207⟩ Physical block replacement Further, the physical block address and physical block erase count described in the spare block management table 32 of the spare block SB1 selected in step S203, and the physical block of the logical block address LA1 in the logical address management table 31 Exchange address and physical block erase count. At the time of replacement, for example, a predetermined storage area (not shown) in the RAM 30 may be used as a temporary storage area. As a result, the physical block entered in the logical address management table 31 and the physical block entered in the spare block management table 32 are replaced. After the replacement, the spare block management table 32 is sorted according to the size of the physical block erase count (data erase count).

[0085] ⟨Step S208⟩ Updating the Logical Address Group Index Since the number of logical address writes in the logical address management table 31 is incremented in step S206, it may be necessary to update the logical address group index. Specifically, the ratio (0 to 1) between the maximum number of logical address writes and the number of logical address writes for all logical block addresses is calculated, for example, 0.25, 0.5, 0.75 The four indexes A, B, C, and D are classified again with three threshold values. If there is a logical block address with an index different from the previous one, the logical address group column of the logical block address in FIG. 6 is updated to that index.

[0086] ⟨Step S209⟩ Updating the spare block group index In step S207, the physical block address and physical block erase count described in the spare block management table 32 are rewritten and sorted according to the magnitude of the physical block erase count. The block group index needs to be updated. Specifically, a ratio (0 to 1) between the maximum number of physical block erase times of all spare blocks and the number of physical block erase times is calculated, for example, 3 of 0.25, 0.5, and 0.75. The four indexes A, B, C, and D are classified again with one threshold. Then, the spare block group column of all physical block addresses in the spare block management table 34 is updated to the index.

[0087] Steps S201 to S209 are repeated each time an instruction to write data to an arbitrary logical address is received from the host 3 to the memory controller 10.

[0088] As described above, in this embodiment, in the memory system in which the relationship of logical-physical conversion changes every time data is rewritten, the frequency of writing is counted as an attribute of the logical block address, and the relative magnitude relationship of the frequency is calculated. The logical block addresses are grouped, and the physical blocks constituting the spare block are also grouped according to the relative size of the erase count.

[0089] When rewriting data, the logical address group with a low write frequency is combined with a spare block group with a high erase frequency, and a logical address group with a high write frequency is combined with a spare block group with a low erase frequency. A spare block is selected for the logical address requested to be rewritten in the combination.

[0090] As a result, the phenomenon of selecting a physical block with a small number of write / erase times as a spare block at a logical address with a low rewrite frequency, which has occurred in the conventional passive wear leveling, can be avoided. The purpose of wear leveling, which is averaging, can be realized with higher accuracy, and the life of the memory system can be extended compared to the conventional case. Further, as in

the first embodiment, it is possible to expect a longer life of the memory system by reducing the required number of active wear leveling.

[0091] In the present embodiment, the number of spare block groups has been described as being the same as the number of logical address groups. However, the relationship between the number of times of writing and the size of the number of times of erasing the spare block are reversed as logical address attributes. The number is not necessarily the same as long as the purpose of group combination can be realized.

[0092] Further, a simpler implementation example of the present embodiment may be as follows. That is, the ratio of the number of writes for each logical block address to the maximum value of the number of logical address writes is classified into A, B, and C logical address groups from the larger, for example, in the magnitude relationship between the threshold values 0.1 and 0.9. The ratio of the number of physical block erases to the maximum number of physical block erases for all spare blocks, for example, the spare block groups of A, B, and C from the smaller one in the magnitude relationship between the threshold values 0.1 and 0.9. And groups with the same index may be associated with each other.

[0093] (Third Embodiment) A block diagram showing a configuration of a memory system according to a third embodiment of the present invention is the same as FIG. 1, but this embodiment relates to active wear leveling. . The RAM 30 includes a logical address management table 31 shown in FIG. 3 and a spare block management table 32 shown in FIG. The description of the operations common to the first embodiment of each component is omitted.

[0094] Also in the present embodiment, as shown in FIG. 2 as an example, logical-physical conversion in which one physical block address, that is, one physical block of the memory unit 40 is associated with one logical block address is performed. To be done. However, the logical-physical conversion relationship is not limited as long as it does not depart from the gist of the present embodiment. Further, in response to a data write instruction from the host 3 to an arbitrary logical address, a spare block (having no logical address) is selected, data is written, and associated with the logical address. That is, the place where the relation of logical-physical conversion changes each time data is rewritten is the same as in the first and second embodiments.

[0095] In the present embodiment, in active wear leveling, which is the physical block rewrite frequency averaging control when there is no rewrite instruction from the host 3, the physical block rewrite frequency is small and the logical block address write frequency is small. Replace the physical block with a spare block with high fatigue level. However, active wear leveling is not executed for a physical block with a large number of writes to a logical block address even if the number of rewrites of the physical block is small. As a result, the accuracy of averaging the number of rewrites of the physical block can be improved as compared with the conventional case.

[0096] That is, in the active wear leveling, the rewrite frequency of the logical block address of the physical block in which valid data is written is considered. Specifically, the memory controller 10 performs active wear leveling in consideration of the number of logical address writes in the logical address management table 31 of FIG.

[0097] A specific procedure of active wear leveling performed by the memory controller 10 of the present embodiment will be described below using the flowchart of FIG.

[0098] ⟨Step S301⟩ When there is no instruction to write data to the logical address from the start condition determination host 3, the CPU 20 performs the physical block erase count of the logical address management table 31 and the physical block erase count of the spare block management table 32 in the RAM 30. Check whether or not the physical block erase count imbalance exceeds a predetermined condition, such as when the difference between the maximum and minimum erase counts in all physical blocks exceeds a predetermined threshold. Judge as the start condition. Moreover, it is good also as starting periodically or determining start conditions for every fixed time. If the start condition is satisfied, the process proceeds to step S302. If not, step S301 is repeated.

[0099] ⟨Step S302⟩ Logical address write count threshold determination CPU 20 accesses the logical address management table 31 in the RAM 30, and satisfies the condition that the logical address write count in the logical address management table 31 is smaller than the first threshold. Determine if there is any. The first threshold only needs to be held in the RAM 30 or the like. As the value of the first threshold, for example, the

maximum value of the number of logical address writes for all the logical block addresses shown in FIG. You may enable it to select the thing with a small logical address write count.

[0100] However, since the purpose of the first threshold is to select a logical block address that is predicted to have few future write requests from the host 3, the present invention is not limited to the above example and coefficient, and is not a relative value but a specific value, for example. It may be a numerical value of the number of times of writing. If there is no logical block address satisfying the condition, the process returns to step S301. If there is a logical block address satisfying the condition, the process proceeds to step S303.

[0101] ⟨Step S303⟩ Physical Block Erase Frequency Threshold Determination The CPU 20 is smaller than the second threshold value among the physical block erase times of the row of the logical block address satisfying the condition of Step S302 in the logical address management table 31 in the RAM 30. It is determined whether there is one satisfying the condition. The second threshold only needs to be held in the RAM 30 or the like. As the value, for example, the physical block in which valid data is written by setting the maximum value of the physical block erase count for all the logical block addresses in FIG. 3 to a coefficient such as 0.1 to 0.3. A block that can select a block having a smaller physical block erase count may be selected.

[0102] Since the purpose is to select a physical block having a lower physical fatigue level from the physical blocks corresponding to the logical block address satisfying the condition of step S302, the second threshold is not limited to the above example or coefficient. A specific physical block erase count may be used. If there is no physical block that satisfies the condition, the process returns to step S301. If there is a physical block that satisfies the condition, the physical block is a vacant physical block, and the corresponding logical block address is a vacant logical address. Then, the process proceeds to step S304.

[0103] ⟨Step S304⟩ The fatigue erasure block erase / data write spare block selection processing unit 23 determines that the physical block erase count in the spare block shown in the spare block management table 32 of FIG. Select the fatigue spare block that is larger or the maximum value. Then, after the block of the fatigue spare block is erased, the data written in the vacant physical block is written.

[0104] The third threshold is basically a condition that it is larger than the second threshold, but a constant may be selected as long as it is based on the physical absolute fatigue level of the physical block. However, a relative value may be selected, for example, a value obtained by multiplying the maximum value of the physical block erase count for all spare blocks shown in FIG. 4 by a coefficient such as 0.9. As long as a spare block with a high degree of fatigue can be selected, the third threshold value is not limited to the above example or coefficient.

[0105] Further, if there is one off-the-shelf physical block that satisfies the condition of step S303, a fatigue spare block having the maximum value in the spare block management table 32 for the physical block erase count may be selected. When there are multiple free physical blocks, it is more effective if you select the reverse order so that the data is written to the fatigue spare block with the smaller number of logical address writes corresponding to it, and the physical block erase number is larger. Wear leveling.

[0106] In addition, if the spare block having the physical block erase count larger than the second threshold value is selected as the fatigue spare block and the data written in the vacant physical block is written, the active wear leveling is realized. The fatigue spare block may not be selected in consideration of the threshold value. Next, the process proceeds to step S305.

[0107] ⟨Step S305⟩ Incrementing the number of times of erasure of the fatigue spare block The erasure number counting unit 22 in the CPU 20 counts the block erasure in step S304 for each physical block address, and the physical block in the spare block management table 32 of FIG. Increase the physical block erase count of the address by one.

[0108] ⟨Step S306⟩ Replacement spare block selection processing unit 23 of the spare physical block and the fatigue spare block The physical block address and the physical block erasure count of the fatigue spare block that has written the data written in the spare physical block in step S304, The physical block address and physical block erasure count of the relevant physical block in the logical address management table 31 are exchanged. At the time of replacement, for example, a predetermined storage area (not shown) or the like in the RAM 30 may be used as a temporary storage area.

[0109] As a result, the physical block address of the unoccupied physical block is erased from the logical address management table 31 and newly entered in the spare block management table 32, and the unoccupied physical block becomes a new spare block. After the replacement, the spare block management table 32 is sorted according to the size of the physical block erase count (data erase count).

[0110] When a data write instruction is not received from the host 3 to the memory controller 10, steps 301 to 306 are performed.

[0111] As described above, in the present embodiment, in active wear leveling when there is no rewrite instruction from the host, the physical block of the physical block is limited to the physical block of the logical block address with a low write request frequency from the host. When the number of times of erasure is small, replace with a spare block with high fatigue. As a result, the active replacement of a physical block with a logical block address, which frequently requires write requests from the host, with a spare block having a high degree of fatigue based only on the condition that the number of erasures of the physical block is low. Wear leveling can be avoided.

[0112] In other words, a logical address with a high data rewrite frequency other than the above-described busy logical address is not subjected to active wear leveling even if the number of rewrites of the physical block corresponding to the address is small. This is because even if such physical block is not intended for active wear leveling, write requests from the host should come frequently, and at that time, the spare block and physical block are forcibly exchanged. It is.

[0113] Therefore, according to the present embodiment, it is possible to reduce the required number of active wear leveling while maintaining the accuracy of the physical block rewrite frequency averaging by the conventional active wear leveling. Therefore, it is possible to reduce the number of physical block erasures associated with voluntary data rewriting of active wear leveling, so that the life of the memory system can be expected to be extended.

[0114] In addition, as described in step S304, when there are a plurality of off-the-shelf physical blocks, data is written to a fatigue spare block having a smaller physical block erase count as the corresponding logical address write count is smaller. By selecting in reverse order, it is possible to increase the accuracy of the physical block rewrite frequency averaging for the same reason as described in the first and second embodiments. That is, according to the present embodiment, it is possible to improve the accuracy of the physical block rewrite frequency averaging by the active wear leveling, and it is possible to further extend the life of the memory system.

[0115] Furthermore, since this embodiment improves active wear leveling, it can be used in combination with the improved passive wear leveling described in the first and second embodiments, and further effects are expected by the combined use. it can. That is, when there is a rewrite instruction from the host, the memory controller 10 may operate according to the flowchart of FIG. 5 or FIG. 12, and when there is no rewrite instruction from the host, according to the flowchart of FIG. As a result, it is expected that the accuracy of erasing the physical block is more accurately averaged than in the first to third embodiments, and the life of the memory system can be further extended.

[0116] In any of the above-described embodiments, the number of writes to the logical address in a storage device in which a plurality of storage elements having deterioration characteristics depending on the number of physical rewrites are mounted and the correspondence of logical-physical conversion changes with each rewrite Is based on the idea that this will be a predicted value of the frequency of writing to the logical address in the future. In other words, spare blocks with a high number of erasures and a high degree of fatigue are intentionally assigned to those with a low number of writes to logical addresses, and storage is performed so that the physical rewrite times of storage elements can be averaged more accurately in the future. The device has a longer life.

[0117] Therefore, the present invention is not limited to the above-described embodiment, and various modifications can be made without departing from the scope of the invention in the implementation stage. Further, the above embodiments include inventions at various stages, and various inventions can be extracted by appropriately combining a plurality of disclosed constituent requirements. For example, even if some constituent requirements are deleted from all the constituent requirements shown in the embodiment, the problem described in the column of the problem to be solved by the invention can be solved, and is described in the column of the effect of the invention. When an effect is obtained, a configuration from which this configuration requirement is

deleted can be extracted as an invention. Furthermore, the constituent elements over different embodiments may be appropriately combined.

[0118] 2 Semiconductor memory device, 3 host, 10 memory controller, 11 bus, 12 host I / F (interface), 13 ROM, 14 error correction section, 15 NAND I / F, 20 CPU, 21 write count coefficient section, 22 erase count Counting unit, 23 Spare block selection processing unit, 30 RAM, 31 Logical address management table, 32 Spare block management table, 40 Non-volatile semiconductor memory unit, 41 Memory cell, 42 Word line, 50, 51 Logical block address, 60, 61 Physical block address, 70 logical-physical conversion table, S101 to S107, S201 to S209, S301 to S306.

(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2011-203916
(P2011-203916A)
(43)公開日　平成23年10月13日(2011.10.13)

(51)Int.Cl.	F I	テーマコード（参考）
G06F 12/16 (2006.01)	G06F 12/16 310A	5B018

審査請求 未請求　請求項の数 8　ＯＬ　（全 21 頁）

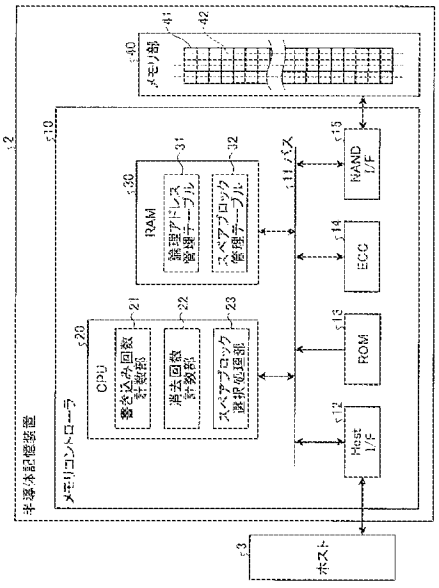
(21)出願番号　特願2010-69418(P2010-69418)
(22)出願日　平成22年3月25日(2010.3.25)

(71)出願人　000003078
株式会社東芝
東京都港区芝浦一丁目１番１号
(74)代理人　100089118
弁理士　酒井　宏明
(72)発明者　助川　博
東京都港区芝浦一丁目１番１号　株式会社東芝内

Ｆターム(参考)　5B018 GA04　HA23　NA06　QA15

(54)【発明の名称】メモリコントローラ、および半導体記憶装置

(57)【要約】
【課題】ブロック消去回数平均化をより高精度に行うメモリコントローラを提供する。
【解決手段】ブロックを複数有する不揮発性メモリ４０の書換え制御を行うメモリコントローラ１０であって、ホスト３が指定するブロック単位の論理アドレスである論理ブロックアドレスとブロックのメモリ４０での記憶位置を示す物理ブロックアドレスとの対応を記憶する第１管理テーブル３１と、ホスト３からメモリ４０への書き込み回数を論理ブロックアドレス毎に記憶する第２管理テーブル３１と、物理ブロックアドレス毎の消去回数を記憶する第３管理テーブル３１、３２と、ホスト３からの論理アドレスを指定した書き込み要求に対して、対応する論理アドレスをもたないスペアブロックを選択して書き込む制御部２３とを備え、論理ブロックアドレス毎に記憶される書き込み回数と物理ブロックアドレス毎に記憶される消去回数に基づいて、消去回数を複数ブロック間で平均化する。
【選択図】図１



【特許請求の範囲】
【請求項１】
データ消去の単位であるブロックを複数個有する不揮発性半導体メモリのデータを書き換えるための制御を行うメモリコントローラであって、
ホストが指定する前記ブロック単位の論理アドレスとしての論理ブロックアドレスと、前記ブロックの前記不揮発性半導体メモリでの記憶位置を示す物理ブロックアドレスとの対応を記憶する第１の管理テーブルと、
前記ホストから前記不揮発性半導体メモリへのデータの書き込み回数を前記論理ブロックアドレス毎に記憶する第２の管理テーブルと、
前記物理ブロックアドレス毎のデータ消去回数を記憶する第３の管理テーブルと、
前記ホストからの論理アドレスを指定した書き込み要求に対して、対応する論理アドレスをもたない前記ブロックであるスペアブロックを選択し、当該スペアブロックにデータを書き込む書き込み制御部とを備え、
前記論理ブロックアドレス毎に記憶されるデータの書き込み回数と前記物理ブロックアドレス毎に記憶されるデータ消去回数に基づいて、データ消去回数を複数の前記ブロックの間で平均化することを特徴とするメモリコントローラ。
【請求項２】
前記書き込み制御部は、
前記第２の管理テーブルに記憶されている書き込み回数が相対的に多い論理ブロックアドレスを指定した書き込み要求に対しては前記第３の管理テーブルに記憶されている消去回数が前記スペアブロックの中で相対的に少ないスペアブロックを選択し、
前記第２の管理テーブルに記憶されている書き込み回数が相対的に少ない論理ブロックアドレスを指定した書き込み要求に対しては前記第３の管理テーブルに記憶されている消去回数が前記スペアブロックの中で相対的に多いスペアブロックを選択する
ことを特徴とする請求項１に記載のメモリコントローラ。
【請求項３】
前記第２の管理テーブルおよび前記第３の管理テーブルに基づき、前記書き込み回数の大小関係と前記消去回数の大小関係が逆になるように前記論理ブロックアドレスと前記スペアブロックを対応づけ、
前記書き込み制御部は、前記ホストからの論理アドレスを指定した書き込み要求に対して、前記対応づけに基づいてスペアブロックを選択する
ことを特徴とする請求項１に記載のメモリコントローラ。
【請求項４】
前記第２の管理テーブルに記憶された書き込み回数の大小関係に基づいて前記論理ブロックアドレスを複数の論理アドレスグループに分類し、前記第３の管理テーブルに記憶された消去回数の大小関係に基づいて前記スペアブロックを複数のスペアブロックグループに分類し、前記書き込み回数の大小関係と前記消去回数の大小関係が逆になるように前記論理アドレスグループと前記スペアブロックグループを対応づけ、
前記書き込み制御部は、前記ホストからの論理アドレスを指定した書き込み要求に対して、当該論理アドレスが分類されている前記論理アドレスグループに対応づけられている前記スペアブロックグループに分類されているスペアブロックを選択する
ことを特徴とする請求項１に記載のメモリコントローラ。
【請求項５】
前記書き込み回数の大小関係に基づく前記論理ブロックアドレスの分類は、前記第２の管理テーブルに記憶されている前記書き込み回数の最大値に対する前記書き込み回数の比を閾値と比較することにより行い、
前記消去回数の大小関係に基づく前記スペアブロックの分類は、前記第３の管理テーブルに記憶されている前記消去回数の前記スペアブロックの中での最大値に対する前記消去回数の比を閾値と比較することにより行う
ことを特徴とする請求項４に記載のメモリコントローラ。

【請求項６】
前記書き込み制御部は、前記ホストからの書き込み要求が無いときに、前記第２の管理テーブルに記憶された書き込み回数が第１閾値より小さい論理ブロックアドレスに前記第１の管理テーブルで対応する物理ブロックアドレスの前記第３の管理テーブルに記憶された消去回数が第２閾値より小さい条件を満たす場合に、当該条件を満たす論理ブロックアドレスである有閑論理アドレスに対応する物理ブロックアドレスが示す有閑物理ブロックに書き込まれているデータを、前記スペアブロックの中から前記第３の管理テーブルに記憶された消去回数が第３閾値よりも大きい疲労スペアブロックを選択して書き込み、
前記疲労スペアブロックの物理ブロックアドレスを前記有閑論理アドレスに対応するように前記第１の管理テーブルに書き込んで前記有閑物理ブロックの物理ブロックアドレスを消去することにより当該有閑物理ブロックを前記スペアブロックにする
ことを特徴とする請求項１に記載のメモリコントローラ。
【請求項７】
前記第１閾値は、前記第２の管理テーブルに記憶された書き込み回数の最大値に所定の定数を乗じた値である
ことを特徴とする請求項６に記載のメモリコントローラ。
【請求項８】
請求項１から請求項７のいずれか１項に記載のメモリコントローラと、前記不揮発性半導体メモリとを具備することを特徴とする半導体記憶装置。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、メモリコントローラ、および半導体記憶装置に関する。
【背景技術】
【０００２】
ＮＡＮＤ型フラッシュメモリは、ブロックと呼ばれる複数のメモリセルに記憶されたデータを一括して消去処理を行うことにより構造を簡略化し、安価でかつ大容量を実現している。そしてさらに、ＮＡＮＤ型フラッシュメモリは、可動部がなく低消費電力であることから、ＨＤＤに代わる記憶装置、携帯電話または携帯音楽プレーヤ等のホストの記憶装置として広く利用されている。しかし、ＮＡＮＤ型フラッシュメモリには書き込み／消去回数に制限があることが知られている。
【先行技術文献】
【特許文献】
【０００３】
【特許文献１】特開２００４－３２６５２３号公報
【発明の概要】
【発明が解決しようとする課題】
【０００４】
本発明は、ブロックの消去回数の平均化を高い精度で行うことができるメモリコントローラ、および半導体記憶装置を提供する。
【課題を解決するための手段】
【０００５】
本願発明の一態様によれば、データ消去の単位であるブロックを複数個有する不揮発性半導体メモリのデータを書き換えるための制御を行うメモリコントローラであって、ホストが指定する前記ブロック単位の論理アドレスとしての論理ブロックアドレスと、前記ブロックの前記不揮発性半導体メモリでの記憶位置を示す物理ブロックアドレスとの対応を記憶する第１の管理テーブルと、前記ホストから前記不揮発性半導体メモリへのデータの書き込み回数を前記論理ブロックアドレス毎に記憶する第２の管理テーブルと、前記物理ブロックアドレス毎のデータ消去回数を記憶する第３の管理テーブルと、前記ホストからの論理アドレスを指定した書き込み要求に対して、対応する論理アドレスをもたない前記

ブロックであるスペアブロックを選択し、当該スペアブロックにデータを書き込む書き込み制御部とを備え、前記論理ブロックアドレス毎に記憶されるデータの書き込み回数と前記物理ブロックアドレス毎に記憶されるデータ消去回数に基づいて、データ消去回数を複数の前記ブロックの間で平均化することを特徴とするメモリコントローラが提供される。

【発明の効果】

【０００６】

本発明によれば、ブロックの消去回数の平均化を高い精度で行うことができるメモリコントローラ、および半導体記憶装置を提供できる。

【図面の簡単な説明】

【０００７】

【図１】図１は、第１乃至３の実施の形態のメモリシステムの構成を示すブロック図である。

【図２】図２は、第１乃至３の実施の形態の半導体記憶装置における物理アドレスと論理アドレスとの関係を示す図である。

【図３】図３は、第１及び第３の実施の形態の論理アドレス管理テーブルを示す図である。

【図４】図４は、第１及び第３の実施の形態のスペアブロック管理テーブルを示す図である。

【図５】図５は、第１の実施の形態のメモリコントローラが行う受動的ウェアレベリングの具体的な手順を示すフローチャートである。

【図６】図６は、第２の実施の形態の論理アドレス管理テーブルを示す図である。

【図７】図７は、論理アドレス毎の書き込み回数の分布を示す図である。

【図８】図８は、図７を書き込み回数の大小関係でグループ分けした図である。

【図９】図９は、第２の実施の形態のスペアブロック管理テーブルを示す図である。

【図１０】図１０は、スペアブロック毎のブロック消去の回数の分布を示す図である。

【図１１】図１１は、図１０をブロック消去の回数の大小関係でグループ分けした図である。

【図１２】図１２は、第２の実施の形態のメモリコントローラが行う受動的ウェアレベリングの具体的な手順を示すフローチャートである。

【図１３】図１３は、第３の実施の形態のメモリコントローラが行う能動的ウェアレベリングの具体的な手順を示すフローチャートである。

【発明を実施するための形態】

【０００８】

ＮＡＮＤ型フラッシュメモリのメモリセルに対する書き込み／消去回数の制限は、基板に対してゲートに高電圧がかけられフローティングゲートに電子が注入されること（書き込み）、及び、ゲートに対してソースに高電圧がかけられフローティングゲートから電子が引き抜かれること（消去）に起因する。即ち、書き込み／消去処理が特定のメモリセルで何回も実行されるとフローティングゲート周りの酸化膜が劣化し、データが破壊されてしまうことがある。

【０００９】

このような書き込み／消去処理が特定のメモリセルに集中することを回避するために、メモリコントローラが書き込み／消去処理回数の平均化をはかっている。これは、メモリコントローラが、消去処理の単位である物理ブロックの消去回数をカウントし消去処理回数の多い物理ブロックと少ない物理ブロックとを入れ替えて、書き込み／消去処理回数の平均化をはかる、いわゆるウェアレベリングによって実施されている。

【００１０】

しかしながら、従来のウェアレベリングにおいては、物理ブロックの属性としての書き込み／消去回数しか参照していなかった（例えば、特許文献１参照。）。特許文献１では、最も書き込み回数が少ない物理ブロックと書き込み回数が所定のチェック回数に達した物理ブロックとを交換するブロック交換手段を備えることにより書き込み回数の平均化を

試みている。

【００１１】

ウェアレベリングを行う半導体記憶装置では一般に、論理アドレスを物理アドレスに変換するための論理アドレス／物理アドレス変換（以下、「論物変換」という。）テーブルが用いられている。この論物変換テーブルを用いることによって、ホストが論理アドレスで指定する更新位置に偏りがあった場合でも、半導体記憶装置内部における物理的な記憶位置を分散させることが可能となる。ウェアレベリングを行う半導体記憶装置では、このように書き換えの都度、論理アドレス／物理アドレスの対応関係が変化する。

【００１２】

ＳＳＤ（Solid State Drive）等のウェアレベリングが行われている半導体記憶装置のように、書き換えの都度、論理アドレス／物理アドレスの対応関係が変わる場合において、ホストからの外部要求としての書き換え頻度の偏りは、物理ブロックの属性ではなく、本来論理アドレスの属性である。しかしながら従来のウェアレベリングにおいては、上記したように物理ブロックの属性としての書き込み／消去回数しか参照されておらず、論理アドレスの属性としての書き換え頻度の偏り特性が全く考慮されていなかった。

【００１３】

従って、従来のウェアレベリングにおいては、例えば、ホストからの外部要求としての書き換え頻度が少ない論理アドレスに対して、書き込み／消去回数が少ない物理ブロックの物理アドレスを割り当てる可能性があった。この場合、その物理ブロックは割り当て後の新しい論物変換の関係において更新される可能性が低く、書き込み／消去回数が少ないままである可能性が高く、結果として物理ブロックの書き換え回数の平均化が適切に行われないという問題があった。

【００１４】

以下に添付図面を参照して、本発明の実施の形態にかかるメモリコントローラ、半導体記憶装置を詳細に説明する。なお、これらの実施の形態により本発明が限定されるものではない。

【００１５】

（第１の実施の形態）

図１は、本発明の第１の実施の形態にかかるメモリシステムとしての半導体記憶装置２の構成を示すブロック図である。半導体記憶装置２は、パソコンまたはデジタルカメラ等のホスト３と着脱可能に接続されるメモリカード、または、ホスト３の内部に収納され、ホスト３の外部記憶装置として機能するエンベデッドタイプのメモリ等である。

【００１６】

半導体記憶装置２は、不揮発性半導体メモリ（以下、単に「メモリ部」ともいう。）４０と、メモリコントローラ１０とを有する。メモリ部４０は例えばＮＡＮＤ型フラッシュメモリであり、単位セルである多数のメモリセル４１が、ビット線（図示せず）およびワード線４２の交点にマトリクス状に配置された構造を有している。メモリ部４０におけるデータ消去は、単位セル複数個からなる物理ブロック単位で行われ、メモリ部４０は複数の物理ブロックを備えている。メモリ部４０に対する書き込み／読み出しは、物理ページ単位に行われる。１つの物理ブロックは複数の物理ページから構成されるので、物理ページのサイズは物理ブロックのサイズより小さい。

【００１７】

メモリコントローラ１０は、制御部であるＣＰＵ２０と、ＲＡＭ３０と、ホスト　Ｉ／Ｆ（インターフェイス）１２と、ＲＯＭ１３と、記憶するデータの符号化処理および記憶されたデータの復号化処理を行う誤り訂正（ＥＣＣ：Error Correcting Code）回路１４と、ＮＡＮＤ　Ｉ／Ｆ（インターフェイス）１５とを備え、これらがバス１１を介して接続されている。

【００１８】

メモリコントローラ１０は、ＣＰＵ２０による制御に従って、ホストＩ／Ｆ１２を介してホスト３とＲＡＭ３０との間でデータ送受信を行い、ＮＡＮＤ　Ｉ／Ｆ１５を介してメ

モリ部４０とＲＡＭ３０との間でデータ送受信を行う。ＣＰＵ２０は、ホスト３とＲＡＭ３０との間のデータ送受信制御、メモリ部４０とＲＡＭ３０との間のデータ送受信制御を行う通常制御部の他に、メモリ部４０へのデータの書き込み回数を論理アドレス毎に計数する書き込み回数計数部２１、メモリ部４０のデータ消去の単位である物理ブロック毎のデータ書き換え回数としてデータ消去回数を計数する消去回数計数部２２、対応する論理アドレスが割り当てられていない物理ブロックであるスペアブロックの選択を行うスペアブロック選択処理部２３を備える。スペアブロック（フリーブロック）は、内部に有効データを含まない、用途未割り当ての物理ブロックである。

【００１９】

スペアブロック選択処理部２３は、ホスト３からデータの書き込み要求が発生すると、バス１１を介してＲＡＭ３０にアクセスして、そこに保持された管理情報（例えば、後述するスペアブロック管理テーブル）などに基づいて、データを書き込むスペアブロックの選択を行う。

【００２０】

図２は、本実施の形態における論物変換の概念を示すものであるが、その前提となる用語を説明する。論理アドレスとは、ホストが使用するアドレスであり、例えばＬＢＡ（Logical Block Addressing）である。ＬＢＡは、セクタ（サイズ：例えば５１２Ｂ）に対して０からの通し番号をつけた論理アドレスである。セクタは前述した物理ページよりも小さい単位である。物理アドレスとは、メモリ部４０での記憶位置を示すアドレスである。

【００２１】

本実施の形態では、データ消去の単位である物理ブロックのサイズと同一の区分サイズとなる論理アドレスの単位として、上記論理アドレス（例えばＬＢＡ）の上位アドレスである論理ブロックアドレスを用いて、物理ブロックの物理アドレスである物理ブロックアドレスに対応付ける。即ち図２に示すように、１個の論理ブロックアドレスに対して１個の物理ブロックアドレスが対応付けられる論物変換が行われることとする。しかし、本実施の形態の要旨に含まれる技術思想の範囲であるならば、この論物変換関係に必ずしも限定されるわけではない。

【００２２】

ＲＡＭ３０は、ホスト３とメモリ部４０との間に介在するキャッシュ領域として機能する領域の他に、論理アドレス管理テーブル３１及びスペアブロック管理テーブル３２を備えている。論理アドレス管理テーブル３１は、図３に示すように論理ブロックアドレスをメモリ部４０における物理ブロックアドレスに変換する第１の管理テーブルである論理アドレス／物理アドレス変換テーブル（論物変換テーブル）７０を含んでいる。

【００２３】

上述したように、スペアブロック選択処理部２３はホスト３からのデータの書き込み指示に対して、対応する論理アドレスが割り当てられていないスペアブロックを選択するので、本実施の形態においては、データ書き換えの都度、論物変換の関係が変わることになる。ホスト３からのメモリ部４０に対する書き込み指示には、通常、先頭論理アドレス、データサイズが含まれている。

【００２４】

さらに、論理アドレス管理テーブル３１は、図３に示すように上記論物変換テーブル７０に登録された物理ブロックアドレスが示す物理ブロックのデータ消去回数（物理ブロック消去回数）、及び論理ブロックアドレス毎にホスト３から指示された書き込み回数（論理アドレス書き込み回数）を記憶している。

【００２５】

従って、論理アドレス管理テーブル３１は論理ブロックアドレス毎のホスト３からの書き込み回数を記憶する第２の管理テーブルである論理書き込み回数テーブルも含んでいることになる。論理書き込み回数テーブルは、論理アドレスの属性としての書き換え頻度の偏り特性を反映したものとなる。

【００２６】

スペアブロック管理テーブル３２は、図４に示すように、対応する論理アドレスが割り当てられていない物理ブロックであるスペアブロックの物理ブロックアドレスと、それに対応する物理ブロックのデータ消去回数（物理ブロック消去回数）とを記憶している。このテーブルは、例えば図４に示すように、物理ブロック消去回数の大小に従って物理ブロックアドレスがソートされていてもよい。

【００２７】

従って、論理アドレス管理テーブル３１に登録された物理ブロックアドレス毎の物理ブロックのデータ消去回数の情報と、スペアブロック管理テーブル３２に登録された物理ブロックアドレス毎の物理ブロックのデータ消去回数の情報とを合わせると、物理ブロックアドレス毎のデータ消去回数を記憶する第３の管理テーブルである物理消去回数テーブルが構成される。物理消去回数テーブルは、物理ブロック毎の消去（及び書き込み）回数の頻度、即ち当該物理ブロックの物理的な疲労度を反映したものとなる。

【００２８】

ＣＰＵ２０は、論理アドレス管理テーブル３１に含まれる論物変換テーブル７０にもとづいて論理アドレスと物理アドレスとの変換を行うアドレス変換部（図示せず）としての動作をファームウエア（ＦＷ）により実行する。ホスト３からのコマンド入力に応じた半導体記憶装置２全体の制御もＣＰＵ２０においてファームウエアで実行される。

【００２９】

ＲＯＭ１３は、半導体記憶装置２のブートプログラム等が格納されている記憶部であり、半導体メモリ部３０の一部または図示しない不揮発性記憶部にもメモリコントローラ１０が半導体記憶装置２を制御するための情報が記憶される。

【００３０】

本実施の形態においては、メモリコントローラ１０が備えるＣＰＵ２０は、ホスト３から任意の論理アドレスへのデータの書き込み指示があった場合、書き込み回数計数部２１が論理ブロックアドレス毎に計数している論理アドレス書き込み回数をインクリメントする。その結果は、バス１１を介してＲＡＭ３０内の論理アドレス管理テーブル３１の論理アドレス書き込み回数の欄に記憶される。

【００３１】

また、ＣＰＵ２０は、メモリ部４０の各物理ブロックにおいてデータの消去があった場合も、消去回数計数部２２が物理ブロックアドレス毎に計数している物理ブロック消去回数をインクリメントする。その結果は、当該物理ブロックが対応する論理アドレス（即ち、論理ブロックアドレス）を有している場合は、バス１１を介してＲＡＭ３０内の論理アドレス管理テーブル３１の当該物理ブロックの物理ブロックアドレスが登録された行（即ち、上記論理ブロックアドレスが登録された行）の物理ブロック消去回数の欄に記憶される。

【００３２】

データ消去のあった物理ブロックが、対応する論理アドレスが割り当てられていない物理ブロックであるスペアブロックの場合は、バス１１を介してＲＡＭ３０内のスペアブロック管理テーブル３２の当該物理ブロックの物理ブロックアドレスが記憶された行の物理ブロック消去回数の欄に記憶される。

【００３３】

従来、ホスト３からの書き込み要求時に行われていたウェアレベリング（以下、受動的ウェアレベリングと呼ぶ）は、ホスト３からの書き込み要求で指定された論理アドレスへの過去の書き込み（書き換え）要求回数は全く考慮されずに行われていた。即ち、従来の受動的ウェアレベリングにおいては、スペアブロックの消去回数のみに依存して、例えば、図４のスペアブロック管理テーブル３２において物理ブロック消去回数が少ないスペアブロックを選択して、書き込み（または、ブロック消去及び書き込み）を行い、当該物理ブロックにホストから指定された論理アドレスを割り当てていた。

【００３４】

しかし、このような受動的ウェアレベリングでは、消去回数の平均化というウェアレベ

リングの目的が適切に実現出来ない可能性があった。例えば、ホストからの外部要求としての書き換え頻度が少ない論理アドレスに偶々書き換え要求が来た場合に、消去回数が少ない物理ブロックをスペアブロックから選択してデータを書き込む可能性があった。この場合、その物理ブロックは当該論理アドレス割り当て後の新しい論物変換の関係においても消去回数が少ないままである可能性が高く、結果として物理ブロックの書き換え回数の偏りが生じてしまい、平均化が適切に行われないという問題があった。

【００３５】

そこで、本実施の形態においては、ホストからデータの書き込み（書き換え）指示があった論理アドレスの書き込み頻度が少ない場合は、当該論理アドレスに対しては、あえて消去回数が大きいスペアブロックを選択して当該論理アドレスを割り当て、選択したスペアブロックに当該データを書き込む。逆に、ホストから書き込み指示があった論理アドレスの書き込み頻度が大きい場合は、当該論理アドレスに対しては、消去回数が少ないスペアブロックを選択して当該論理アドレスを割り当て、選択したスペアブロックに当該データを書き込む。これにより、上述した平均化が適切に行われないという問題を解決することが可能となる。

【００３６】

具体的には、スペアブロック選択処理部２３（書き込み制御部）が、論理アドレス管理テーブル３１に記憶されている書き込み回数が相対的に多い論理ブロックアドレスを指定した書き込み要求に対しては、スペアブロック管理テーブル３２に記憶されている消去回数が相対的に少ないスペアブロックを選択し、書き込み回数が相対的に少ない論理アドレスを指定した書き込み要求に対しては、消去回数が相対的に多いスペアブロックを選択する。

【００３７】

例えば、ホスト３から図３の論理ブロックアドレス「ｎ－１」を指定した書き込み要求が来た場合は、論理アドレス管理テーブル３１内での論理アドレス書き込み回数「１５０」が相対的に少ないと判断して、スペアブロック管理テーブル３２内での物理ブロック消去回数が相対的に多い図４の物理ブロックアドレス「ＷＷＷ」のスペアブロックを選択してデータを書き込む。逆に、図３の論理ブロックアドレス「ｎ」を指定した書き込み要求が来た場合は、論理アドレス管理テーブル３１内での論理アドレス書き込み回数「４００」が相対的に多いと判断して、スペアブロック管理テーブル３２内での物理ブロック消去回数が相対的に少ない図４の物理ブロックアドレス「ａａａ」のスペアブロックを選択してデータを書き込む。

【００３８】

更に具体的には、図３の論理アドレス管理テーブル３１の論理アドレス書き込み回数の大小関係でソートした論理ブロックアドレスと、図４のスペアブロック管理テーブル３２の物理ブロック消去回数の大小関係でソートしたスペアブロックの物理ブロックアドレスを、両者の大小関係が逆になるように対応づけておいてもよい。

【００３９】

また、大小関係を逆順に対応づける方法としては、上記のように生の回数値を用いるのでない方法で行うこともできる。例えば、図３の論理アドレス管理テーブル３１にさらに「書き込み回数相対値」の欄を設け、各論理アドレス書き込み回数をその時点での論理アドレス書き込み回数の最大値で割った値（０～１）を論理ブロックアドレス毎に保持しておく。さらに、スペアブロック管理テーブル３２にも「消去回数相対値」の欄を設け、各物理ブロック消去回数をその時点でのスペアブロックの中での物理ブロック消去回数の最大値で割った値（０～１）をスペアブロック毎に保持しておく。

【００４０】

そして、ホスト３からの論理ブロックアドレスを指定した書き込み要求に対して、図３の当該論理ブロックアドレスの「書き込み回数相対値」がＸであった場合。例えば、以下の（式１）に従い、「好適消去回数相対値」Ｙを算出する。

【００４１】

Ｙ＝１－Ｘ　　（式１）

【００４２】

スペアブロック選択処理部２３は、ここで求まったＹに最も近い「消去回数相対値」を有するスペアブロックをスペアブロック管理テーブル３２から選択してデータを書き込む。例えば、「書き込み回数相対値」Ｘ＝０．２のように書き込み回数が相対的に小さい論理ブロックアドレスを指定した書き込み要求に対しては、「好適消去回数相対値」Ｙ＝０．８に最も近い消去回数がスペアブロックの中で相対的に大きいスペアブロックを選択する。

【００４３】

上記（式１）はＸとＹが逆順の関係になるような式であれば、上記の式に限定されない。また、図７に示すホストからの論理アドレス毎の書き込み回数頻度の偏りと図１０に示すスペアブロック毎のブロック消去回数頻度の偏りを反映させた非線形な式を用いて対応づけがより均等になるように工夫しても構わない。

【００４４】

以下に、図５のフローチャートを用いて、本実施の形態のメモリコントローラ１０が行う受動的ウェアレベリングの具体的な手順を示す。

【００４５】

＜ステップＳ１０１＞　データ書き込み指示

ホスト３からメモリコントローラ１０に、例えば論理ブロックアドレスＬＡ１を指定したデータ書き込み要求が来る。

【００４６】

＜ステップＳ１０２＞　スペアブロックの選択

スペアブロック選択処理部２３は、指定された論理ブロックアドレスＬＡ１に対して、上記（式１）に基づいてスペアブロックＳＢ１を選択する。

【００４７】

＜ステップＳ１０３＞　ブロック消去及びデータの書き込み

ステップＳ１０２で選択したスペアブロックＳＢ１に相当するメモリ部４０内の物理ブロックアドレスの物理ブロックをブロック消去し、その物理ブロックにホスト３から指示のあったデータの書き込みを行う。

【００４８】

＜ステップＳ１０４＞　物理ブロック消去回数のインクリメント

ＣＰＵ２０内の消去回数計数部２２は、ステップＳ１０３におけるブロック消去を物理ブロックアドレス毎に計数しており、図４のスペアブロック管理テーブル３２の当該物理ブロックアドレスの物理ブロック消去回数を１増やす。

【００４９】

＜ステップＳ１０５＞　論理アドレス書き込み回数のインクリメント

ＣＰＵ２０内の書き込み回数計数部２１は、ステップＳ１０１におけるホスト３からの各論理ブロックアドレスへの書き込み指示を論理ブロックアドレス毎に計数しており、図３の論理アドレス管理テーブル３１の論理ブロックアドレスＬＡ１の論理アドレス書き込み回数を１増やす。

【００５０】

＜ステップＳ１０６＞　物理ブロックの交換

さらにステップＳ１０２で選択したスペアブロックＳＢ１のスペアブロック管理テーブル３２に記載された物理ブロックアドレス及び物理ブロック消去回数と、論理アドレス管理テーブル３１の論理ブロックアドレスＬＡ１の物理ブロックアドレス及び物理ブロック消去回数とを交換する。交換時に例えば、ＲＡＭ３０内の所定の記憶領域（図示せず）等を一時記憶領域として利用しても構わない。これにより論理アドレス管理テーブル３１にエントリされていた物理ブロックとスペアブロック管理テーブル３２にエントリされていた物理ブロックが入れ替えられたことになる。交換後、スペアブロック管理テーブル３２を物理ブロック消去回数（データ消去回数）の大小に従ってソートする。

【００５１】
＜ステップＳ１０７＞　テーブルソート
スペアブロック管理テーブル３２を物理ブロック消去回数（データ消去回数）の大小関係に従ってソートする。
【００５２】
ホスト３からメモリコントローラ１０に、任意の論理アドレスへのデータ書き込み指示が来る度にステップＳ１０１～Ｓ１０７を繰り返す。
【００５３】
以上説明したように本実施の形態においては、ホストからデータの書き込み指示があった論理アドレスの書き込み頻度が少ない場合は、あえて消去回数が大きいスペアブロックを選択して当該データを書き込み、ホストから書き込み指示があった論理アドレスの書き込み頻度が大きい場合は、消去回数が少ないスペアブロックを選択して当該データを書き込むようにする。
【００５４】
このように、本実施の形態に係る半導体記憶装置２は、データ書き換えの都度、論物変換の関係が変わる場合において、論理アドレスの属性である書き換え回数、即ち論理アドレスの書き換え頻度を参照して、物理ブロックの書き換え回数の平均化を行う。
【００５５】
本実施の形態では、受動的ウェアレベリングにおいて、書き込み頻度が少ない論理ブロックアドレスには消去回数が大きく物理的な疲労の程度が大きい物理（スペア）ブロックを割り当て、逆に、書き込み頻度が大きい論理ブロックアドレスには消去回数が少ない物理的な疲労の程度が小さい物理（スペア）ブロックを割り当てることによって、物理ブロックの書き換え回数の平均化の精度を高めることが可能になると考えられる。
【００５６】
即ち、従来の受動的ウェアレベリングにおいて生じていた、書き換え頻度が少ない論理アドレスに書き込み／消去回数が少ない物理ブロックをスペアブロックとして選択する現象を回避することができるので、物理ブロックの書き換え回数の平均化というウェアレベリングの目的をより精度を高めて実現することが可能となり、メモリシステムの寿命を従来に比べて延ばすことが可能となる。
【００５７】
一方、ホストからのデータ書き換え指示が無いときであっても、消去回数が少ない物理ブロックに書き込まれているデータを消去回数が多いスペアブロックにメモリシステムが自主的に書き移すことにより、当該消去回数が少ない物理ブロックをスペアブロックにするウェアレベリング（以下、能動的ウェアレベリングと呼ぶ）の方法がある。
【００５８】
上記実施の形態においては、受動的ウェアレベリングの物理ブロック消去回数平均化が高精度になることによって、能動的ウェアレベリングの必要回数も低減されると考えられる。従って、能動的ウェアレベリングにおけるメモリシステムの自主的なデータ書き換えに伴う物理ブロックの消去の回数を減らすことも可能になるので、これによるメモリシステムの長寿命化も期待できる。
【００５９】
（第２の実施の形態）
本発明の第２の実施の形態にかかるメモリシステムとしての半導体記憶装置２の構成を示すブロック図は図１と同様である。同じ構成要素の説明は省略する。本実施の形態においても、例として図２に示すように１個の論理ブロックアドレスに対して１個の物理ブロックアドレス、即ち、メモリ部４０の１つの物理ブロックが対応付けられた論物変換が行われることとする。しかし、本実施の形態の要旨を逸脱しない範囲であるならば、この論物変換関係に限定されるわけではない。
【００６０】
また、ホスト３から任意の論理アドレスへのデータの書き込み指示に対して（論理アド

レスが割り当てられていない）スペアブロックを選択してデータを書き込んで当該論理アドレスと対応づける。即ち、データ書き換えの都度、論物変換の関係が変わるところも、第１の実施の形態と同様である。

【００６１】

本実施の形態においては、スペアブロック選択処理部２３の処理と、ＲＡＭ３０内に備えられる論理アドレス管理テーブル３１及びスペアブロック管理テーブル３２の構成が第１の実施の形態とは異なる。

【００６２】

図６に示す本実施の形態における論理アドレス管理テーブル３１は、第１の実施の形態と同様に、論理ブロックアドレスを対応する物理ブロックのメモリ部４０における物理アドレスである物理ブロックアドレスに変換する論理アドレス／物理アドレス変換テーブル（論物変換テーブル）７０を含んでいる。論物変換テーブル７０に記載された物理ブロックアドレスのデータ消去回数（物理ブロック消去回数）を記憶し、論理ブロックアドレス毎の書き込み回数（論理アドレス書き込み回数）を記憶して論理書き込み回数テーブルを含んでいることも第１の実施の形態と同様である。

【００６３】

しかし図６に示すように、本実施の形態の論理アドレス管理テーブル３１には、論理ブロックアドレス毎に論理アドレスグループのインデックスがさらに記憶されている。一般に、ホスト３からの論理アドレス毎の書き込み回数頻度には図７に示すように偏りが存在する。従って、図６の論理アドレス書き込み回数の大小関係に基づいて論理ブロックアドレス全てを複数の論理アドレスグループのいずれかに分類することが可能である。例えば、図８に示すようにＡ～Ｄのインデックスをそれぞれ持った４つの論理アドレスグループに分類することができる。

【００６４】

具体的には、例えば図６に示された全ての論理ブロックアドレスに対する論理アドレス書き込み回数の最大値に対する各論理ブロックアドレス書き込み回数の比（０～１）を、３つの閾値、例えば、０．２５、０．５、０．７５との大小関係で４つに分類し、図８に示すように比が大きい方から、Ａ、Ｂ、Ｃ、Ｄのインデックスをそれぞれ持った４つ論理アドレスグループに分類することができる。

【００６５】

論理アドレスグループの数は複数であればよく４に限定されるわけではない。また、論理アドレス書き込み回数の大小関係に基づいて、全ての論理ブロックアドレスをほぼ同数ずつの論理ブロックアドレスを有する複数の論理アドレスグループに分類してもよく、論理アドレス書き込み回数の大小関係に基づいた分類ならば閾値の間隔を含め上記分類方法に限定されない。このように定めたインデックスを図６の論理アドレス管理テーブル３１の論理アドレスグループの欄に各論理ブロックアドレスの属性として、論理ブロックアドレス毎に記憶させる。

【００６６】

一方図９に示す本実施の形態におけるスペアブロック管理テーブル３２も、第１の実施の形態と同様に、対応する論理アドレスが割り当てられていない物理ブロックであるスペアブロックの物理ブロックアドレスとそれに対応する物理ブロックの物理ブロック消去回数（データ消去回数）を記憶している。そして、物理ブロック消去回数の大小に従って物理ブロックアドレスがソートされている。

【００６７】

しかし図９に示すように、本実施の形態のスペアブロック管理テーブル３２には、スペアブロックの物理ブロックアドレス毎にスペアブロックグループのインデックスがさらに記憶されている。

【００６８】

一般に、スペアブロック毎のブロック消去の回数頻度には図１０に示すように偏りが存在する。従って、図９の物理ブロック消去回数の大小関係に基づいてスペアブロック全て

を、上記論理アドレスグループの数と例えば同数のスペアブロックグループのいずれかに分類することが可能である。例えば、図１１に示すようにＡ～Ｄのインデックスをそれぞれ持った４つのスペアブロックグループに分類することができる。

【００６９】

具体的には、例えば図９に示された全てのスペアブロックに対する物理ブロック消去回数の最大値に対する各物理ブロック消去回数の比（０～１）を、３つの閾値、例えば、０．２５、０．５、０．７５との大小関係で４つに分類し、図１１に示すように比が小さい方から、Ａ、Ｂ、Ｃ、Ｄのインデックスをそれぞれ持ったスペアブロックグループに分類することができる。

【００７０】

スペアブロックグループの数は上記論理アドレスグループの数と、例えば同数であればよく４に限定されるわけではない。また、物理ブロック消去回数の大小関係に基づいて、全てのスペアブロックをほぼ同数ずつのスペアブロックを有する複数のスペアブロックグループに分類してもよく、物理ブロック消去回数の大小関係に基づいた分類ならば閾値の間隔を含め上記分類方法に限定されない。このインデックスを図９のスペアブロック管理テーブル３２のスペアブロックグループの欄に、スペアブロックの物理ブロックアドレス毎に記憶させる。

【００７１】

本実施の形態においては、ホストからデータの書き込み（書き換え）指示があった論理ブロックアドレスが書き込み頻度の少ない論理アドレスグループに属している場合は、当該論理ブロックアドレスに対しては、消去回数の頻度が大きいスペアブロックグループに属するスペアブロックを選択して当該論理ブロックアドレスを割り当て、選択したスペアブロックに相当するメモリ部４０内の物理ブロックに当該データを書き込む。

【００７２】

逆に、ホストから書き込み指示があった論理ブロックアドレスが書き込み頻度の大きい論理アドレスグループに属している場合は、当該論理ブロックアドレスに対しては、消去回数の頻度が少ないスペアブロックグループに属するスペアブロックを選択して当該論理ブロックアドレスを割り当て、選択したスペアブロックに相当するメモリ部４０内の物理ブロックに当該データを書き込む。

【００７３】

上述した図８及び図１１の例の場合、論理アドレス書き込み回数の大小関係と物理ブロック消去回数の大小関係が逆になるように、それぞれの論理アドレスグループには同じインデックスのスペアブロックグループ、すなわち、Ａ同士、Ｂ同士、Ｃ同士、Ｄ同士を組み合わせる。この組み合わせの中から、書き込み指示があった論理アドレスに対してスペアブロックの選択を行う。

【００７４】

以下に、図１２のフローチャートを用いて、本実施の形態のメモリコントローラ１０が行う受動的ウェアレベリングの具体的な手順を示す。

【００７５】

＜ステップＳ２０１＞　データ書き込み指示

ホスト３からメモリコントローラ１０に、例えば論理ブロックアドレスＬＡ１を指定したデータ書き込み要求が来る。

【００７６】

＜ステップＳ２０２＞　論理アドレスグループの判定

スペアブロック選択処理部２３は、図６の論理アドレス管理テーブル３１にアクセスし論理ブロックアドレスＬＡ１の論理アドレスグループのインデックスに基づき、論理アドレスグループを判定する。

【００７７】

＜ステップＳ２０３＞　スペアブロックの選択

スペアブロック選択処理部２３は、図９のスペアブロック管理テーブル３２にアクセス

して、ステップＳ２０２で判定した論理アドレスグループと同じインデックスのスペアブロックグループに属するスペアブロックＳＢ１を選択する。

【００７８】

上述したように、それぞれの論理アドレスグループには論理アドレス書き込み回数の大小関係と物理ブロック消去回数の大小関係が逆になるようにスペアブロックグループが対応付けられている。従って、ステップＳ２０２で判定した論理アドレスグループに対応するスペアブロックグループに属するスペアブロックであれば、どれを選択しても、例えば当該スペアブロックグループに属するスペアブロックをランダムに１つ選択するようにしても構わない。

【００７９】

しかし、第１の実施の形態の考えをここでも反映させて、ステップＳ２０２で判定した論理アドレスグループの中での論理ブロックアドレスＬＡ１の論理アドレス書き込み回数の相対的な大小の程度に応じて、同一インデックスのスペアブロックグループからスペアブロックＳＢ１を選択するようにした方が、物理ブロックの消去回数の平均化がさらに効果的に行える。即ち、ステップＳ２０２で判定した論理アドレスグループの中で論理アドレスＬＡ１の論理アドレス書き込み回数が相対的に大きい場合は、対応するスペアブロックグループに属するスペアブロックの中で物理ブロック消去回数が相対的に小さいスペアブロックを選択し、逆に論理アドレス書き込み回数が相対的に大きい場合は、物理ブロック消去回数が相対的に小さいスペアブロックを選択してもよい。

【００８０】

このような機構は、例えば第１の実施の形態で説明した（式１）等を用いれば実現できる。この場合、図６の論理アドレス管理テーブル３１の論理アドレスグループの欄に論理アドレスグループのインデックスに付加して、当該インデックスの論理アドレスグループ内での書き込み回数の最大値でその行の論理アドレス書き込み回数を割った値である論理アドレスグループ毎の「書き込み回数相対値」をさらに記憶させる。例えば「Ａ：０．３」等である。同様に図９のスペアブロック管理テーブル３２のスペアブロックグループの欄にも、各スペアブロックグループ内での消去回数の最大値でその行の物理ブロック消去回数を割った値スペアブロックグループ毎の「消去回数相対値」を「Ｄ：０．４」のように付加記憶させる。そしてステップＳ２０２で判定した論理アドレスグループと同じインデックスのスペアブロックグループを選択した後、その中で（式１）等に従って「好適消去回数相対値」に最も近いスペアブロックグループＳＢ１を選択する。

【００８１】

＜ステップＳ２０４＞　ブロック消去及びデータの書き込み

ステップＳ２０３で選択したスペアブロックＳＢ１に相当するメモリ部４０内の物理ブロックアドレスの物理ブロックをブロック消去し、その物理ブロックにホスト３から指示のあったデータの書き込みを行う。

【００８２】

＜ステップＳ２０５＞　物理ブロック消去回数のインクリメント

ＣＰＵ２０内の消去回数計数部２２は、ステップＳ２０４におけるブロック消去を物理ブロックアドレス毎に計数しており、図９のスペアブロック管理テーブル３２の当該物理ブロックアドレスの物理ブロック消去回数を１増やす。

【００８３】

＜ステップＳ２０６＞　論理アドレス書き込み回数のインクリメント

ＣＰＵ２０内の書き込み回数計数部２１は、ステップＳ２０１におけるホスト３からの各論理ブロックアドレスへの書き込み指示を論理ブロックアドレス毎に計数しており、図６の論理アドレス管理テーブル３１の論理ブロックアドレスＬＡ１の論理アドレス書き込み回数を１増やす。

【００８４】

＜ステップＳ２０７＞　物理ブロックの交換

さらにステップＳ２０３で選択したスペアブロックＳＢ１のスペアブロック管理テーブ

ル３２に記載された物理ブロックアドレス及び物理ブロック消去回数と、論理アドレス管理テーブル３１の論理ブロックアドレスＬＡ１の物理ブロックアドレス及び物理ブロック消去回数を交換する。交換時に例えば、ＲＡＭ３０内の所定の記憶領域（図示せず）等を一時記憶領域として利用しても構わない。これにより論理アドレス管理テーブル３１にエントリされていた物理ブロックとスペアブロック管理テーブル３２にエントリされていた物理ブロックが入れ替えられたことになる。交換後、スペアブロック管理テーブル３２を物理ブロック消去回数（データ消去回数）の大小に従ってソートする。

【００８５】

＜ステップＳ２０８＞　論理アドレスグループのインデックスの更新

ステップＳ２０６で論理アドレス管理テーブル３１の論理アドレス書き込み回数がインクリメントされているので、論理アドレスグループのインデックスの更新の必要が生じる場合がある。具体的には、全ての論理ブロックアドレスに対する論理アドレス書き込み回数の最大値と各論理アドレス書き込み回数との比（０～１）を計算し、例えば、０．２５、０．５、０．７５の３つの閾値で再度４つのインデックスＡ、Ｂ、Ｃ、Ｄに分類する。以前と異なるインデックスになった論理ブロックアドレスがあった場合は、図６の当該論理ブロックアドレスの論理アドレスグループの欄をそのインデックスに更新する。

【００８６】

＜ステップＳ２０９＞　スペアブロックグループのインデックスの更新

ステップＳ２０７で、スペアブロック管理テーブル３２に記載された物理ブロックアドレス及び物理ブロック消去回数が書き換えられ、物理ブロック消去回数の大小に従ってソートされているので、スペアブロックグループのインデックスの更新の必要が生じる。具体的には、全てのスペアブロックの物理ブロック消去回数の最大値と各物理ブロック消去回数との比（０～１）を計算し、例えば、０．２５、０．５、０．７５の３つの閾値で再度４つのインデックスＡ、Ｂ、Ｃ、Ｄに分類する。そして、スペアブロック管理テーブル３４の全ての物理ブロックアドレスのスペアブロックグループの欄をそのインデックスに更新する。

【００８７】

ホスト３からメモリコントローラ１０に、任意の論理アドレスへのデータ書き込み指示が来る度にステップＳ２０１～Ｓ２０９を繰り返す。

【００８８】

以上説明したように本実施の形態においては、データ書き換えの都度、論物変換の関係が変わるメモリシステムにおいて、論理ブロックアドレスの属性として書き込み回数頻度を計数してその頻度の相対的な大小関係で論理ブロックアドレスをグループ分けし、スペアブロックを構成する物理ブロックも消去回数の相対的な大小関係でグループ分けしておく。

【００８９】

論物変換の関係が変わるデータ書き換え時に、書き込み回数頻度が小さい論理アドレスグループは消去回数が大きいスペアブロックグループと、書き込み回数頻度が大きい論理アドレスグループは消去回数が小さいスペアブロックグループと組み合わせて、その組み合わせの中で書き換え要請があった論理アドレスにスペアブロックを選択する。

【００９０】

これにより、従来の受動的ウェアレベリングにおいて生じていた、書き換え頻度が少ない論理アドレスに書き込み／消去回数が少ない物理ブロックをスペアブロックとして選択する現象を回避することができるので、物理ブロックの書き換え回数の平均化というウェアレベリングの目的をより精度を高めて実現することが可能となり、メモリシステムの寿命を従来に比べて延ばすことが可能となる。また、第１の実施の形態同様、能動的ウェアレベリングの必要回数の低減によるメモリシステムの長寿命化も期待できる。

【００９１】

なお、本実施の形態においては、スペアブロックグループの数は論理アドレスグループの数と同数として説明したが、論理アドレスの属性として書き込み回数の大小関係とスペ

アブロックの消去回数の大小関係が逆順となるグループの組み合わせにする趣旨が実現できるのであれば、必ずしも同数でなくてもかまわない。

【００９２】

また、本実施の形態のより簡易な実現例として以下のようにしてもよい。即ち、論理アドレス書き込み回数の最大値に対する各論理ブロックアドレス毎の書き込み回数の比を、例えば閾値０．１と０．９との大小関係により大きい方からＡ、Ｂ、Ｃの論理アドレスグループに分類し、全てのスペアブロックに対する物理ブロック消去回数の最大値に対する各物理ブロック消去回数の比も、例えば閾値０．１と０．９との大小関係により小さい方からＡ、Ｂ、Ｃのスペアブロックグループに分類して同一インデックスのグループ同士を対応づけてもよい。

【００９３】

（第３の実施の形態）

本発明の第３の実施の形態にかかるメモリシステムの構成を示すブロック図は、図１と同様であるが、本実施の形態は能動的ウェアレベリングに関するものである。ＲＡＭ３０は、図３に示す論理アドレス管理テーブル３１と、図４に示すスペアブロック管理テーブル３２を備えている。各構成要素の第１の実施の形態と共通な動作の説明は省略する。

【００９４】

本実施の形態においても、例として図２に示すように１個の論理ブロックアドレスに対して１個の物理ブロックアドレス、即ち、メモリ部４０の１つの物理ブロックが対応付けられた論物変換が行われることとする。しかし、本実施の形態の要旨を逸脱しない範囲であるならば、この論物変換関係に限定されるわけではない。また、ホスト３から任意の論理アドレスへのデータの書き込み指示に対して（論理アドレスを持たない）スペアブロックを選択してデータを書き込んで当該論理アドレスと対応づける。即ち、データ書き換えの都度、論物変換の関係が変わるところも、第１及び第２の実施の形態と同様である。

【００９５】

本実施の形態においては、ホスト３からの書き換え指示が無いときの物理ブロックの書き換え回数平均化制御である能動的ウェアレベリングにおいて、物理ブロックの書き換え回数が少なく且つ論理ブロックアドレスへの書き込み回数が少ない物理ブロックを疲労度が高いスペアブロックと交換する。しかし、物理ブロックの書き換え回数が少なくても論理ブロックアドレスへの書き込み回数が多い物理ブロックに対しては能動的ウェアレベリングを実行しない。これにより、物理ブロックの書き換え回数平均化の精度を従来に比べて高めることが可能となる。

【００９６】

即ち、能動的ウェアレベリングにおいて、有効なデータが書き込まれている物理ブロックの論理ブロックアドレスの書き換え頻度を考慮する。具体的には、メモリコントローラ１０が図３の論理アドレス管理テーブル３１の論理アドレス書き込み回数を考慮に入れた上で能動的ウェアレベリングを行う。

【００９７】

以下に、図１３のフローチャートを用いて、本実施の形態のメモリコントローラ１０が行う能動的ウェアレベリングの具体的な手順を示す。

【００９８】

＜ステップＳ３０１＞　開始条件判定

ホスト３から論理アドレスへのデータの書き込み指示が無いときに、ＣＰＵ２０が、ＲＡＭ３０内の論理アドレス管理テーブル３１の物理ブロック消去回数とスペアブロック管理テーブル３２の物理ブロック消去回数をチェックし、全ての物理ブロックの中での消去回数の最大値と最小値の差が所定の閾値以上になったとき等、物理ブロックの消去回数の不均衡が所定の条件を超えたかどうかを開始条件として判定する。また、一定時間毎に周期的に開始する、或いは開始条件を判定するとしてもよい。開始条件を満たす場合にステップＳ３０２に進む。満たさない場合はステップＳ３０１を繰り返す。

【００９９】

＜ステップＳ３０２＞　論理アドレス書き込み回数の閾値判定
ＣＰＵ２０は、ＲＡＭ３０内の論理アドレス管理テーブル３１にアクセスして、論理アドレス管理テーブル３１の論理アドレス書き込み回数の中で第１閾値より小さいという条件を満たすものがあるかどうかを判定する。第１閾値はＲＡＭ３０等に保持されていればよい。第１閾値の値としては、例えば図３に示された全ての論理ブロックアドレスに対する論理アドレス書き込み回数の最大値に例えば０．１～０．３等の係数を掛けた値にして、相対的に論理アドレス書き込み回数が小さいものを選別出来るようにしてもよい。
【０１００】
しかし、ホスト３からの今後の書き込み要求が少ないことが予測される論理ブロックアドレスを選択するのが第１閾値の目的であるので、上記例や係数に限定されず、相対値ではなく例えば具体的な書き込み回数の数値にしても構わない。条件を満たす論理ブロックアドレスが全く無い場合は、ステップＳ３０１に戻り、条件を満たす論理ブロックアドレスがある場合は、ステップＳ３０３に進む。
【０１０１】
＜ステップＳ３０３＞　物理ブロック消去回数の閾値判定
ＣＰＵ２０は、ＲＡＭ３０内の論理アドレス管理テーブル３１の中でステップＳ３０２の条件を満たした論理ブロックアドレスの行の物理ブロック消去回数の中で第２閾値より小さいという条件を満たすものがあるかどうかを判定する。第２閾値はＲＡＭ３０等に保持されていればよい。その値としては、例えば図３の全ての論理ブロックアドレスに対する物理ブロック消去回数の最大値に例えば０．１～０．３等の係数を掛けた値にして、有効なデータが書き込まれている物理ブロックの中からさらに物理ブロック消去回数が小さいものを選別出来るようなものを選べばよい。
【０１０２】
ステップＳ３０２の条件を満たした論理ブロックアドレスに対応する物理ブロックの中からさらに物理的な疲労度が少ない物理ブロックを選択するのが目的であるので、第２閾値も上記例や係数に限定されず、具体的な物理ブロック消去回数でもかまわない。条件を満たす物理ブロックが全く無い場合は、ステップＳ３０１に戻る。条件を満たす物理ブロックがある場合は、当該物理ブロックを有閑物理ブロックとし、それに対応する論理ブロックアドレスを有閑論理アドレスとする。そしてステップＳ３０４に進む。
【０１０３】
＜ステップＳ３０４＞　疲労スペアブロックのブロック消去及びデータの書き込み
スペアブロック選択処理部２３が、図４のスペアブロック管理テーブル３２に示されたスペアブロックの中での物理ブロック消去回数が第３閾値よりも大きいか或いは最大値である疲労スペアブロックを選択する。そして、当該疲労スペアブロックをブロック消去した上で、上記有閑物理ブロックに書き込まれているデータを書き込む。
【０１０４】
第３閾値としては、基本的に第２閾値より大きいことが条件であるが、物理ブロックの物理的な絶対的な疲労度を基準にするのであれば定数を選択してもよい。しかし、例えば図４に示された全てのスペアブロックに対する物理ブロック消去回数の最大値に例えば０．９等の係数を掛けた値のように相対的な値を選択してもよい。疲労度が高いスペアブロックを選択できるのであれば、第３閾値は上記例や係数に限定されるものではない。
【０１０５】
また、ステップＳ３０３の条件を満たす有閑物理ブロックが１つであれば、物理ブロック消去回数がスペアブロック管理テーブル３２の中で最大値をとる疲労スペアブロックを選択してもよい。有閑物理ブロックが複数ある場合は、その中で対応する論理アドレス書き込み回数がより小さいもの程、物理ブロック消去回数がより大きい疲労スペアブロックにデータを書き込むように逆順に選択すればより効果的な能動的ウェアレベリングとなる。
【０１０６】
また、第２閾値より大きい物理ブロック消去回数のスペアブロックを疲労スペアブロッ

クとして選択して有閑物理ブロックに書き込まれているデータを書き込むのであれば、能動的ウェアレベリングは実現されるので、必ずしも第３閾値を考慮して疲労スペアブロックを選択しなくてもよい。次にステップＳ３０５に進む。

【０１０７】

＜ステップＳ３０５＞　疲労スペアブロックの消去回数のインクリメント

ＣＰＵ２０内の消去回数計数部２２は、ステップＳ３０４におけるブロック消去を物理ブロックアドレス毎に計数しており、図４のスペアブロック管理テーブル３２の当該物理ブロックアドレスの物理ブロック消去回数を１増やす。

【０１０８】

＜ステップＳ３０６＞　有閑物理ブロックと疲労スペアブロックの交換

スペアブロック選択処理部２３は、ステップＳ３０４で有閑物理ブロックに書き込まれているデータを書き込んだ疲労スペアブロックの物理ブロックアドレス及び物理ブロック消去回数と、論理アドレス管理テーブル３１の当該有閑物理ブロックの物理ブロックアドレス及び物理ブロック消去回数を交換する。交換時に例えば、ＲＡＭ３０内の所定の記憶領域（図示せず）等を一時記憶領域として利用しても構わない。

【０１０９】

これにより当該有閑物理ブロックの物理ブロックアドレスが論理アドレス管理テーブル３１から消去されて、スペアブロック管理テーブル３２に新たにエントリされ、当該有閑物理ブロックは新たにスペアブロックになる。交換後、スペアブロック管理テーブル３２を物理ブロック消去回数（データ消去回数）の大小に従ってソートする。

【０１１０】

ホスト３からメモリコントローラ１０に、データ書き込み指示が来ないときに、ステップ３０１～ステップ３０６を行う。

【０１１１】

以上説明したように本実施の形態においては、ホストからの書き換え指示が無いときの能動的ウェアレベリングにおいて、ホストからの書き込み要求頻度が少ない論理ブロックアドレスの物理ブロックに限定して、当該物理ブロックの消去回数が少ない場合に、疲労度が高いスペアブロックと交換する。これにより、従来、ホストからの書き込み要求頻度が多い論理ブロックアドレスの物理ブロックに対しても当該物理ブロックの消去回数が少ないという条件のみに基づいて疲労度が高いスペアブロックと交換していた能動的ウェアレベリングを回避できる。

【０１１２】

即ち、上記有閑論理アドレス以外のデータ書き換え頻度が高い論理アドレスに対しては、例え当該アドレスに対応する物理ブロックの書き換え回数が少なくても能動的ウェアレベリングの対象とはしない。これは、そのような物理ブロックはあえて能動的ウェアレベリングの対象としなくても、ホストからの書き込み要求が頻繁に来るはずなので、そのときに、強制的にスペアブロックと物理ブロックが交換されるからである。

【０１１３】

従って、本実施の形態によって、従来の能動的ウェアレベリングによる物理ブロックの書き換え回数平均化の精度を維持しつつ能動的ウェアレベリングの必要回数を減らすことが可能になる。よって、能動的ウェアレベリングの自主的なデータ書き換えに伴う物理ブロックの消去回数を減らすことが可能になるので、メモリシステムの長寿命化が期待できる。

【０１１４】

また、上記ステップＳ３０４で述べたように、有閑物理ブロックが複数ある場合は、その中で対応する論理アドレス書き込み回数がより小さいもの程、物理ブロック消去回数がより大きい疲労スペアブロックにデータを書き込むように逆順に選択することにより、第１及び２の実施の形態で説明したのと同様な理由により物理ブロックの書き換え回数平均化の精度を高めることが可能となる。即ち、本実施の形態によって能動的ウェアレベリングによる物理ブロックの書き換え回数平均化の精度を高めることも可能であり、メモリシ

ステムのさらなる長寿命化が可能となる。

【０１１５】

さらに、本実施の形態は能動的ウェアレベリングを改良するものであるので、第１及び第２の実施の形態で説明した改良した受動的ウェアレベリングと併用可能であり、併用により更なる効果が期待できる。即ち、ホストからの書き換え指示があったときは、図５或いは図１２のフローチャートに従い、ホストからの書き換え指示がないときは、図１３のフローチャートに従って、メモリコントローラ１０が動作すればよい。これにより、上記した第１乃至３の実施の形態より、物理ブロックの消去回数平均化の精度をさらに高めることが期待でき、メモリシステムの長寿命化がさらに図れることになる。

【０１１６】

上記したいずれの実施の形態も、物理的な書き換え回数に依存した劣化特性を有する記憶素子を複数搭載し、論物変換の対応関係が書き換えの都度変化するストレージデバイスにおいて、論理アドレスへの書き込み回数が、今後の当該論理アドレスへの書き込み頻度の予測値になるという考えに基づいている。即ち、論理アドレスへの書き込み回数が少ないものには消去回数が多く疲労度が高いスペアブロックをあえて割り当て、記憶素子の物理的な書き換え回数の平均化を将来にわたってより高精度に行えるようにしてストレージデバイスの長寿命化を図っている。

【０１１７】

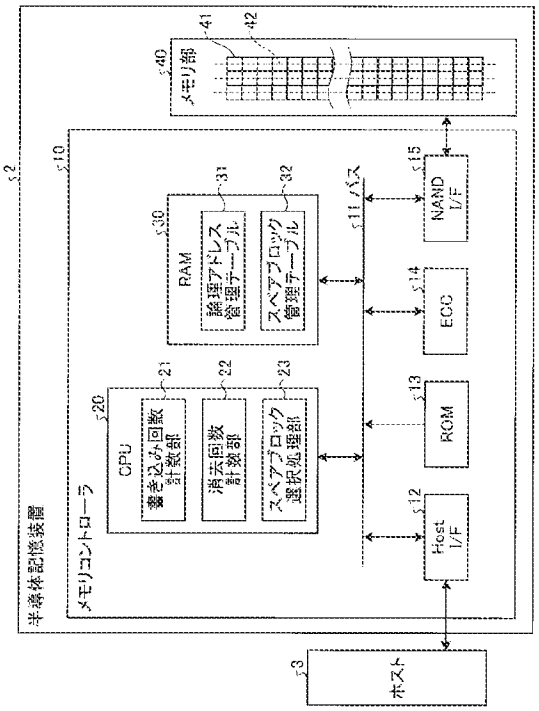
従って、本願発明は上記実施の形態に限定されるものではなく、実施段階ではその要旨を逸脱しない範囲で種々に変形することが可能である。また、上記実施の形態には種々の段階の発明が含まれており、開示される複数の構成要件における適宜な組み合わせにより種々の発明が抽出されうる。例えば、実施の形態に示される全構成要件からいくつかの構成要件が削除されても、発明が解決しようとする課題の欄で述べた課題が解決でき、発明の効果の欄で述べられている効果が得られる場合には、この構成要件が削除された構成が発明として抽出されうる。更に、異なる実施の形態にわたる構成要素を適宜組み合わせてもよい。

【符号の説明】

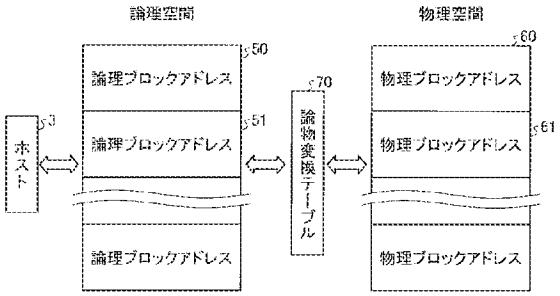
【０１１８】

２　半導体記憶装置、３　ホスト、１０　メモリコントローラ、１１　バス、１２　ホスト　Ｉ／Ｆ（インターフェイス）、１３　ＲＯＭ、１４　誤り訂正部、１５　ＮＡＮＤ　Ｉ／Ｆ、２０　ＣＰＵ、２１　書き込み回数係数部、２２　消去回数計数部、２３　スペアブロック選択処理部、３０　ＲＡＭ、３１　論理アドレス管理テーブル、３２　スペアブロック管理テーブル、４０　不揮発性半導体メモリ部、４１　メモリセル、４２　ワード線、５０、５１　論理ブロックアドレス、６０、６１　物理ブロックアドレス、７０　論物変換テーブル、Ｓ１０１～Ｓ１０７、Ｓ２０１～Ｓ２０９、Ｓ３０１～Ｓ３０６　ステップ。

【図１】



【図２】



【図３】

論理アドレス管理テーブル

論理ブロックアドレス	物理ブロックアドレス	物理ブロック消去回数	論理アドレス書き込み回数
0	XXX	450	200
1	YYY	400	300
.....	.....	.....	.....
n-1	PPP	600	150
n	ZZZ	300	400

70
論物変換テーブル

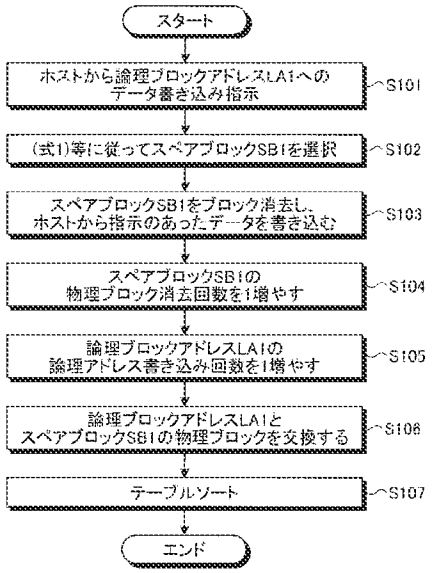
【図４】

スペアブロック管理テーブル

物理ブロックアドレス	物理ブロック消去回数
aaa	300
bbb	350
.....	.....
vvv	700
www	300

小
物理ブロック消去回数
大

【図５】



【図６】

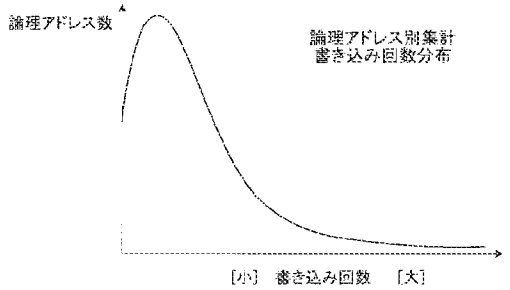
論理アドレス管理テーブル

31

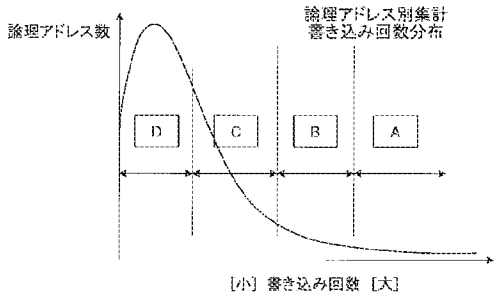
論理ブロックアドレス	物理ブロックアドレス	物理ブロック消去回数	論理アドレス書き込み回数	論理アドレスグループ
0	XXX	450	200	B
1	YYY	400	300	C
⋮	⋮	⋮	⋮	⋮
n-1	PPP	600	150	D
n	ZZZ	300	400	A

70
論物変換テーブル

【図７】



【図８】



【図９】

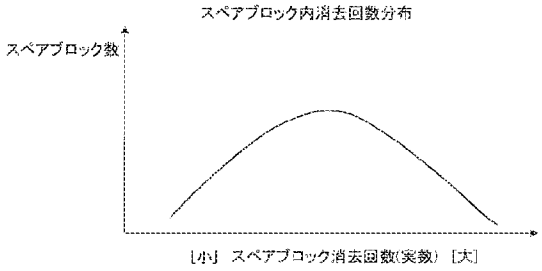
スペアブロック管理テーブル

32

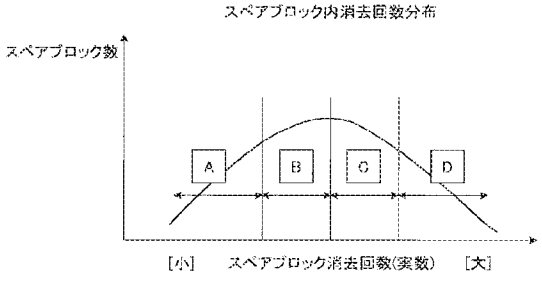
物理ブロックアドレス	物理ブロック消去回数	スペアブロックグループ
aaa	300	A
bbb	350	A
⋮	⋮	⋮
vvv	700	D
www	800	D

小
A
B
C
D
物理ブロック消去回数
大

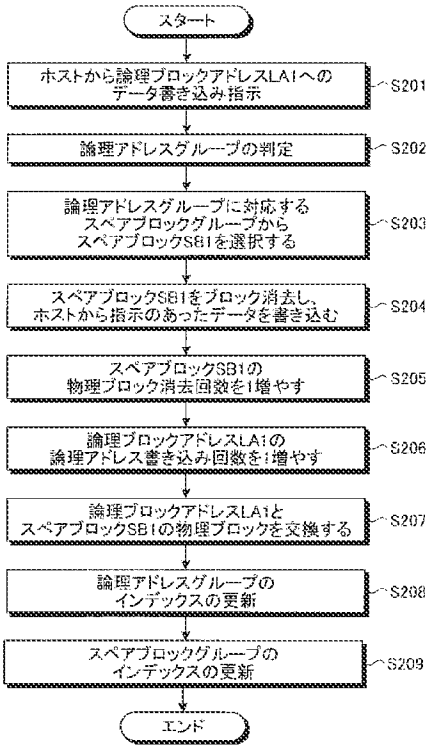
【図１０】



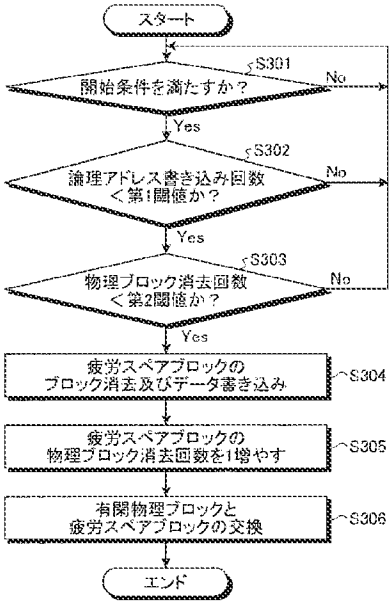
【図１１】



【図１２】



【図１３】


スタート
S301
開始条件を満たすか？
No
Yes
S302
論理アドレス書き込み回数
＜第1閾値か？
No
Yes
S303
物理ブロック消去回数
＜第2閾値か？
No
Yes
疲労スペアブロックの
ブロック消去及びデータ書き込み
S304
疲労スペアブロックの
物理ブロック消去回数を1増やす
S305
有関物理ブロックと
疲労スペアブロックの交換
S306
エンド

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC PAYMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/931,506	12/13/2022 05:03:03 PM ET	RMMS.003C12

Title of Invention

STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	3894	FILED BY	Tayler Amerine
PATENT CENTER #	61292285	AUTHORIZED BY	Aaron Davis
CUSTOMER #	20995	FILING DATE	09/12/2022
CORRESPONDENCE ADDRESS	-	FIRST NAMED INVENTOR	Andrey V. Kuzmin

Payment Information

PAYMENT METHOD	PAYMENT TRANSACTION ID	PAYMENT AUTHORIZED BY
CARD / 2549	E2022BCH04291655	Aimee Kazan
PRE-AUTHORIZED ACCOUNT	**PRE-AUTHORIZED CATEGORY**	
111410	37 CFR 1.16 (National application filing, search, and examination fees); 37 CFR 1.17 (Patent application and reexamination processing fees)	

FEE CODE	DESCRIPTION	ITEM PRICE($)	QUANTITY	ITEM TOTAL($)
2806	SUBMISSION OF AN INFORMATION DISCLOSURE STATEMENT	130.00	1	130.00
			TOTAL AMOUNT:	$130.00

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 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.

Patent Translate
Powered by EPO and Google

Notice

This translation is machine - generated. It cannot be guaranteed that it is intelligible, accurate, complete, reliable or fit for specific purposes. Critical decisions, such as commercially relevant or financial decisions, should not be based on machine - translation output.

DESCRIPTION KR20000025612A

11 Multi - channel voice data control method and apparatus

[0001]

15 The present invention relates to a gateway system, and more particularly, to a method and apparatus for controlling multi - channel voice data in a gateway system between a public switched telephone network and an internet protocol packet network.

18 Recently, as voice call terminals based on Internet Protocol (IP), such as Internet phones, have been commercialized, a gateway system enabling voice calls with existing public telephone networks has been developed and marketed.

21 Typically, a gateway system is divided into a call control conversion part between both networks and a voice data processing conversion part.

23 In particular, since voice data processing conversion must be processed at high speed in real time, it is a key technology for system performance and efficiency.

25 In general, a voice processing part in a gateway system is implemented using a standardized protocol having a structure as shown in FIG. 1 .

27 In such a gateway system, the process of transmitting the voice signal of the public switched telephone network to the LAN network or the Internet network is to rearrange the analog voice signal received from the public switched telephone network into a predetermined format in the public switched telephone network interface 101, and the audio codec (CODEC) 102 When the digital voice data is converted into digital voice data and the multi - channel packet controller 103 periodically packetizes the digital voice data in units of a predetermined amount, a real - time protocol (RTP/RTCP) block 104) and UDP/IP (User Datagram Protocol/Internet Protocol) block 105 adds a UDP header and an RTP header to process a predetermined signal, and then rearranges it in a predetermined format at the LAN interface 106 to connect to the LAN or the Internet. transmit over the

network

39 Conversely, the process of transmitting voice data from a LAN or Internet network to a public switched telephone network is performed through a User Datagram Protocol/Internet Protocol (UDP/IP) block 105 and a real - time The protocol (RTP; Real - Time Protocol) block 104 sequentially interprets the UDP header and the RTP header and controls the corresponding data to transmit voice packet data to the multi - channel packet controller 103, and the multi - channel packet controller (103) separates the voice packet data into individual voice data, converts them into analog voice signals in the audio codec 102, rearranges them into a predetermined format in the public switched telephone network interface 101, and transmits them to the public switched telephone network.

49 In the process of processing/converting voice data in the above, multiple channels must be simultaneously processed in real time.

51 Therefore, since standardized protocols are used to interface between each protocol layer, standardized protocol processing must be satisfied first when exchanging data between each protocol layer, but voice call performance varies depending on how voice data is controlled.

55 In general, the format of voice data between protocol interfaces in a gateway system is the same as the exemplary diagram of FIG. 2 .

57 This illustrates a process in which a voice signal on one line of a public switched telephone network corresponding to one audio codec is relayed to one UDP port.

59 That is, when the digital voice data converted by the audio codec 102 reaches a predetermined number, the multi - channel packet controller 103 bundles them into one packet to generate RTP data, and the real - time protocol (RTP) block 104 generates the above - generated audio data. UDP data is generated by adding an RTP header to RTP data, and UDP header is added to UDP data in the UDP block 105 to be transmitted over the Internet network.

65 Conversely, voice data in the Internet network is sequentially processed in the reverse order and separated into individual voice data in the multi - channel packet controller 103, and then converted into analog voice signals in the audio codec 102 and transmitted to the public switched telephone network. .

69 In this case, as a result of call control at the beginning of one voice call, one public switched telephone network channel is mapped to one port of a UDP block. It is to perform voice data relay between several ports of

72 The real - time protocol (RTP) is a standard protocol for transmitting real - time data such as voice or video in an Internet network, and the RTP header contains control information for controlling RTP data carrying voice data.

75 Meanwhile, since the voice data processing process in the multi - channel packet controller 103 and the real - time protocol (RTP) 104 must be performed at high speed, the implementation structure and scheduling algorithm of both modules influence the overall system performance.

79 However, conventionally, when voice data is transmitted through the Internet

SAMSUNG EX1002, Page 504 of 1031

network, an RTP header is added to the voice data, and then a UDP header is added to the data to which the RTP header is added. There is a problem in that the processing speed is degraded by sequentially performing an operation of decoding the audio data into original voice data.

84 Therefore, in order to improve the conventional problem, the present invention divides the buffer area into voice packet units and then fixedly allocates an RTP header to each area to control the buffer area in a ring method during voice data transmission, thereby real - time protocol (RTP) and a multi - channel audio data control method and apparatus designed to improve processing speed by performing voice data transmission between a block and a multi - channel packet controller.

90 1 is an exemplary diagram showing a protocol structure of a general gateway system;

91 FIG. 2 is an exemplary diagram showing a format of audio data in FIG. 1;

92 3 is an exemplary diagram showing a task scheduling structure of a system for implementing the present invention;

94 4 is a signal flow diagram for the practice of the present invention;

95 5 and 6 are exemplary diagrams showing a buffer allocation structure and process in the present invention.

97 * Description of symbols for main parts of drawings *

98 201 : Timer 202 : Interrupt Service Block

99 203 : Application Program Task Block 204: Multi - Channel Packet Controller

100 205 : UDP block 210a, 210b : memory block

101 220 : receive buffer descriptor 230 : send buffer descriptor

102 In order to achieve the above object, the present invention provides an apparatus and method for implementing a multi - channel packet controller driver and a real - time protocol (RTP) block in one application program task synchronized by a periodically generated timer interrupt. to be

106 In order to achieve the above object, the present invention provides a multi - channel packet controller to control voice packets for each channel for voice packets transmitted from the multi - channel packet controller to a real - time protocol (RTP) block in advance at the time of system initialization. It controls the buffers to which the memory area is fixedly allocated, and the real - time protocol block controls the voice packets for each channel for the voice packets transferred from the real - time protocol (RTP) block to the multi - channel packet controller. It is characterized in that it is configured to control the buffers allocated to the memory area.

114 In the above, when the memory area is fixedly allocated in advance, not only the voice packet but also the fixed RTP header of 12 bytes is allocated to each unit buffer in advance.

117 Hereinafter, the present invention will be described in detail based on the drawings.

118 As shown in the exemplary diagrams of FIGS. 3, 5 and 6, the system for implementing the present invention transmits and receives voice signals between a UDP block 205 that transmits and receives voice packet data with the Internet network and a public telephone network. A multi - channel packet controller 204 that

SAMSUNG EX1002, Page 505 of 1031

performs packetization or packet separation while performing, and a memory block 210a, 210b that stores voice packet data in the UDP block 205 and the multi - channel packet controller 204 and an application program task block 203 that controls voice data transmission between the UDP block 205 and the multi - channel packet controller 204 whenever an interrupt occurs, a timer 201 that periodically generates an interrupt, and the timer It consists of an interrupt service block 202 provided to the application program task block 203 when an interrupt periodically occurs in 201.

130 The application program task block 203 includes a real - time protocol (RTP) and a circuit for driving a multi - channel packet controller.

132 The memory blocks 210a and 210b are configured by pre - allocating a memory area corresponding to a voice packet of each channel in a fixed manner and pre - allocating an RTP header for each memory area during system initialization.

135 The reception buffer descriptor 220 and the transmission buffer descriptor having a flag indicating whether to store voice packets corresponding to the memory areas of the memory blocks 210a and 210b and a fixed address indicating the storage start position of voice packet data. (230) is further provided to configure.

139 The memory areas previously allocated to the memory blocks 210a and 210b are controlled in a ring manner.

141 The operation and effect of the embodiment of the present invention configured as described above will be described.

143 First, a task scheduling process will be described as follows.

144 The application program task block 203 collectively processes data transmitted/received from the UDP block 205 and the multi - channel packet controller 204 in one task in consideration of the characteristics of periodically repeated packet data.

148 In this case, a method of periodically polling is used instead of a method of processing data whenever an interrupt event from the multi - channel packet controller 204 occurs.

151 That is, in the case of an interrupt event driven method from a traffic source, overhead of interrupt service switching time or task switching time is applied to each channel and each packet.

154 For example, if the packetization period is '30msec' and the number of channels is 'N', in the public switched telephone network, 'N' interrupt service switching and task switching must occur on average every '30msec' in the reception direction.

157 In addition, by processing similar to the above for events in the opposite direction, switching overhead due to numerous events within a short time of '30 msec' is applied, and thus process efficiency is greatly reduced.

160 Therefore, in the present invention, when the task scheduling method as shown in FIG. 3 is applied, an interrupt occurs only once every '30 msec' by the periodic interrupt timer 201, and in the interrupt service routine 202, real - time protocol (RTP)/multi - channel packet It is only necessary to transfer the event flag once to

SAMSUNG EX1002, Page 506 of 1031

the application program task block 203, which can simultaneously execute the controller operation.

166 This operation is performed in the same process as the signal flow of FIG. 4 .

167 The suspended application program task block 203 is re - executed whenever a timer interrupt occurs in the timer 201.

169 And, memory blocks 210a and 210b are pre - allocated with memory areas for storing voice packets from the multi - channel packet controller 204 and voice packets from the UDP block 205 during system initialization, and each memory area has The RTP header is also preallocated.

173 Accordingly, the multi - channel packet controller 204 continuously stores digitally converted voice data transmitted from the public switched telephone network in a corresponding area of the memory block 210a, and the UDP block 205 stores the digital voice data transmitted from the Internet network in memory. It is continuously stored in the corresponding area of the block 210b.

178 At this time, when the corresponding memory area of the memory block 210a is full of data, the multi - channel packet controller 204 updates the status bit of the corresponding area of the receive buffer descriptor 220 to '1' and When the memory area is full of data, the UDP block 205 updates the status bit of the corresponding area of the transmission buffer descriptor 230 to '1'.

183 Then, when a periodic interrupt occurs in the timer 201, the interrupt service block 202 provides an event flag notifying the occurrence of a periodic interrupt to the application program task block 203.

186 At this time, the application program task block 203 checks the status bits of the receive buffer descriptor (RxBD) 220 updated by the multi - channel packet controller 204 for each currently active voice communication channel.

189 Accordingly, if the status bit of the corresponding area of the receive buffer descriptor 220 corresponding to the currently active channel is '1' and there is data received from the public switched telephone network, the application program task block 203 adds protocol data to the RTP header. After performing the update, it is transmitted to the UDP block 205 together with the RTP data filled by the multi - channel packet controller 204, and the status bit is updated to '0'.

195 Then, the application program task block 203 checks the port of the UDP block 205 corresponding to the corresponding channel to determine whether data received from the Internet network exists.

198 At this time, if there is a voice packet in the UDP block 205, the application program task block 203 processes the RTP header, copies the voice packet to the corresponding area of the memory block 210b, and transmits a transmit buffer descriptor (TxBD) 230.) updates the status bit of the corresponding area to '1'.

202 Accordingly, the multi - channel packet controller 204 reads the RTP data stored in the corresponding area of the memory block 210b, transmits it to the audio codec, and updates the status bit of the corresponding area of the transmit buffer descriptor 230 to '0'.

SAMSUNG EX1002, Page 507 of 1031

206 The above operations are repeatedly performed for each of the 'N' number of channels, and when all of these operations are completed, the application program task block 203 immediately enters a pause state.

209 That is, the buffer control process in the present invention will be described with reference to FIGS. 5 and 6.

211 The memory blocks 210a and 210b are included in one memory circuit, and in order to have a buffering structure for each voice channel, regions for storing RTP headers and voice packets are continuously and fixedly allocated during system initialization.

215 5 shows a case of storing and transmitting voice packets from a public switched telephone network. The multi - channel packet controller 204 packetizes voice data from each audio codec and stores the data in a corresponding buffer area of the memory block 210a. The RTP data (voice packet) is copied and the status bit of the corresponding area of the receive buffer descriptor 220 is updated.

220 Accordingly, when a periodic interrupt occurs, the application program task block 203, if the status bit of the area corresponding to the currently active channel among the areas of the receive buffer descriptor 220 is '1', UDP including the RTP header and RTP data Send data to UDP block 205 and update its RTP header.

224 In addition, FIG. 6 shows a case of storing and transmitting voice packets from the Internet network. When a periodic interrupt occurs, the application program task block 203 checks the UDP block 205 and, if there is received data, the memory block (After processing the RTP header of the area corresponding to the currently active channel among the areas of 210b) and copying the received voice packet, the status bit of the corresponding area of the transmit buffer descriptor 230 is updated.

230 Accordingly, the multi - channel packet controller 204 checks whether the status bit of the corresponding area of the transmit buffer descriptor 230 is '1', and if it is '1', the voice packet of the corresponding area of the memory block 210b is transmitted to the audio codec. and updates the status bit of the corresponding area of the transmit buffer descriptor 230 to '0'.

235 The above operation is performed simultaneously whenever a periodic interrupt occurs.

237 As described in detail above, the present invention minimizes the load on the microprocessor by the application program task for relaying the voice packets received from the public telephone network and the voice packets received from the Internet network to improve the operation speed. It has the effect of accommodating many channels.

242 In addition, the present invention has the effect of obtaining the same effect by applying to a gateway system between a packet network such as an Internet network and a network transmitting voice in a TDM method such as an integrated information communication network (ISDN) or frame relay in addition to a public telephone network.

SAMSUNG EX1002, Page 508 of 1031

(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(51) Int. Cl.[6]
H04L 12/66

(11) 공개번호 특2000-0025612
(43) 공개일자 2000년05월06일

(21) 출원번호 10-1998-0042765
(22) 출원일자 1998년10월13일

(71) 출원인 엘지전자 주식회사 구자홍
서울특별시 영등포구 여의도동 20번지
(72) 발명자 배성한
서울특별시 서초구 반포동 577-21
(74) 대리인 박장원

심사청구 : 있음

(54) 멀티 채널 음성 데이터 제어 방법 및 장치

요약

본 발명은 멀티채널 음성 데이터 제어 방법 및 장치에 관한 것으로 특히, 버퍼의 영역을 음성 패킷 단위로 분할한 후 각 영역에 RTP 헤더를 고정적으로 할당하여 음성 데이터 전송시 버퍼 영역을 링 방식으로 제어함에 의해 실시간 프로토콜(RTP) 블럭과 멀티 채널 패킷 제어기간의 음성 데이터 전송을 수행함으로써 처리 속도를 향상시키도록 함에 목적이 있다. 이러한 목적의 본 발명은 시스템 초기화시 소정량 단위로 메모리 영역을 미리 할당하고 그 각각의 영역에 RTP 헤더를 미리 할당하는 단계와, 태스크 일시 정지 상태에서 주기적인 인터럽트 발생을 판단하는 단계와, N개의 채널을 순차적으로 검색하여 액티브되는 채널을 판단하는 단계와, 상기에서 액티브되는 채널을 판단하면 해당 채널의 데이터가 존재하는지 판단하는 단계와, 상기에서 해당 채널의 데이터가 존재하면 멀티채널 패킷 제어기로부터의 데이터는 RTP 헤더를 갱신하여 UDP 블럭으로 전송하고 동시에 UDP 블럭으로부터의 데이터는 RTP 헤더를 처리하여 멀티채널 패킷 제어기로 전송하는 단계와, 상기에서 N개의 채널에 대한 데이터 전송이 종료되면 태스크를 일시 정지시키는 단계를 수행한다.

대표도

도4

명세서

도면의 간단한 설명

도 1은 일반적인 게이트웨이 시스템의 프로토콜 구조를 보인 예시도.

도 2는 도1 에서의 음성 데이터의 포맷을 보인 예시도.

도 3은 본 발명의 실시를 위한 시스템의 태스크 스케쥴링 구조를 보인 예시도.

도 4는 본 발명의 실시를 위한 신호 흐름도.

도 5 및 도 6은 본 발명에서의 버퍼 할당 구조 및 처리 과정을 보인 예시도.

* 도면의 주요부분에 대한 부호 설명 *

201 : 타이머 202 : 인터럽트 서비스 블럭
203 : 응용 프로그램 태스크 블럭 204 : 멀티 채널 패킷 제어기
205 : UDP 블럭 210a,210b : 메모리 블럭
220 : 수신 버퍼 디스크립터 230 : 송신 버퍼 디스크립터

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 게이트웨이(GateWay) 시스템에 관한 것으로 특히, 공중 전화망과 인터넷 프로토콜 패킷망간의 게이트웨이 시스템에 있어서 멀티 채널 음성 데이터 제어 방법 및 장치에 관한 것이다.

SAMSUNG EX1002, Page 509 of 1031

최근 인터넷 폰과 같이 인터넷 프로토콜(IP)을 기반으로 하는 음성 통화 단말기가 상용화되면서 기존의 공중 전화망과 음성 통화를 가능하게 해 주는 게이트웨이(Gate-Way) 시스템이 개발되어 시판되고 있다.

통상적으로 게이트웨이 시스템은 양쪽 망간의 호 제어(Call Control) 변환 부분과 음성 데이터 처리 변환 부분으로 구분된다.

특히, 음성 데이터 처리 변환은 실시간으로 고속 처리되어야 하기 때문에 시스템의 성능과 효율성을 위한 핵심 기술이다.

일반적으로 게이트웨이 시스템에서의 음성 처리 부분은 도1 과 같은 구조의 표준화된 프로토콜을 사용하여 구현한다.

이러한 게이트웨이 시스템에서 공중 전화망의 음성 신호가 랜망 또는 인터넷망으로 전송되는 과정은 공중 전화망 인터페이스(101)에서 공중 전화망에서 수신된 아날로그 음성 신호를 소정 포맷으로 재배열하면 오디오 코덱(CODEC)(102)에서 디지털 음성 데이터로 변환하고 멀티 채널 패킷 제어기(103)에서 상기 디지털 음성 데이터를 주기적으로 소정량 단위씩 패킷화하면 실시간 프로토콜(RTP/RTCP ; Real-Time Protocol/Real-Time Transmission Control Protocol) 블럭(104) 및 UDP/IP(User Datagram Protocol/Internet Protocol) 블럭(105)에 의해 UDP 헤더, RTP 헤더를 부가하여 소정 신호 처리된 후 랜 인터페이스(106)에서 소정 포맷으로 재배열되어 랜(LAN) 또는 인터넷망으로 전송한다.

반대로, 랜망 또는 인터넷망에서의 음성 데이터가 공중 전화망으로 전송되는 과정은 디지털 음성 데이터가 랜 인터페이스(106)에서 소정 포맷으로 재열되면 UDP/IP(User Datagram Protocol/Internet Protocol) 블럭(105) 및 실시간 프로토콜(RTP ; Real-Time Protocol) 블럭(104)에서 순차적으로 UDP 헤더, RTP 헤더를 해석하여 해당 데이터를 제어함에 의해 음성 패킷 데이터를 멀티 채털 패킷 제어기(103)으로 전송하고 그 멀티 채널 패킷 제어기(103)가 음성 패킷 데이터를 각각의 음성 데이터로 분리하면 오디오 코덱(102)에서 아날로그 음성 신호로 변환함에 따라 공중 전화망 인터페이스(101)에서 소정 포맷으로 재배열되어 공중 전화망으로 전송된다.

상기에서 음성 데이터의 처리/변환 과정은 실시간으로 동시에 여러 채널을 처리해야 한다.

따라서, 표준화된 프로토콜을 사용하여 각 프로토콜 계층간에 인터페이스하기 때문에 각 프로토콜 계층간에 데이터를 주고 받을 때 표준화된 프로토콜 처리가 우선적으로 만족되어야 하지만 음성 데이터를 어떻게 제어하느냐에 따라 음성 통화의 성능이 달라진다.

일반적으로 게이트웨이 시스템에서 프로토콜 인터페이스간의 음성 데이터 포맷은 도2 의 예시도와 같다.

이는 하나의 오디오 코덱에 대응되는 공중 전화망의 한 회선에서의 음성 신호가 하나의 UDP 포트까지 중계되는 과정을 도시한 것이다.

즉, 오디오 코덱(102)에서 변환되는 디지털 음성 데이터가 소정 개수가 되면 멀티채널 패킷 제어기(103)에서 하나의 패킷으로 묶어 RTP 데이터를 생성하고 실시간 프로토콜(RTP) 블럭(104)에서 상기에서 생성된 RTP 데이터에 RTP 헤더를 부가하여 UDP 데이터를 생성하며 UDP 블럭(105)에서 UDP 데이터에 UDP 헤더를 부가하여 인터넷망으로 전송하게 된다.

반대로, 인터넷망에서의 음성 데이터는 상기와 역순으로 순차 처리됨에 의해 멀티 채널 패킷 제어기(103)에서 각각의 음성 데이터로 분리된 후 오디오 코덱(102)에서 아날로그 음성 신호로 변환되어 공중 전화망으로 전송된다.

이 경우 하나의 음성 통화 초기에 이루어지는 호 제어의 결과에 의해 한 개의 공중전화망 채널은 UDP 블럭의 한 개의 포트와 매핑되는데, 게이트웨이 시스템에서의 다중 채널 음성 처리/전송은 공중전화망의 여러 채널과 UDP 블럭의 여러 포트간의 음성 데이터 중계를 수행하는 것이다.

상기에서 실시간 프로토콜(RTP)은 인터넷망에서 음성 또는 비디오와 같은 실시간성 데이터를 전송하기 위한 표준 프로토콜로서, RTP 헤더에는 음성 데이터가 실려 있는 RTP 데이터를 제어하기 위한 제어 정보들을 담고 있다.

한편, 상기에서 멀티채널 패킷 제어기(103)와 실시간 프로토콜(RTP)(104)에서의 음성 데이터 처리 과정이 고속으로 이루어져야 하기 때문에 양 모듈의 구현 구조 및 스케줄링 알고리즘이 전체 시스템 성능을 좌우한다.

발명이 이루고자하는 기술적 과제

그러나, 종래에는 인터넷망으로 음성 데이터 전송시 음성 데이터에 RTP 헤더를 부가한 후 그 RTP 헤더가 부가된 데이터에 UDP 헤더를 부가하고 반대로 공중 전화망으로 음성 데이터 전송시 UDP 헤더를 제거한 후 RTP 헤더를 제거하여 원래의 음성 데이터로 복호하는 동작을 순차적으로 수행함으로 처리 속도가 저하되는 문제점이 있었다.

따라서, 본 발명은 종래의 문제점을 개선하기 위하여 버퍼의 영역을 음성 패킷 단위로 분할한 후 각 영역에 RTP 헤더를 고정적으로 할당하여 음성 데이터 전송시 버퍼 영역을 링 방식으로 제어함에 의해 실시간 프로토콜(RTP) 블럭과 멀티 채널 패킷 제어기간의 음성 데이터 전송을 수행함으로써 처리 속도를 향상시키도록 창안한 멀티 채널 음성 데이터 제어 방법 및 장치을 제공함에 목적이 있다.

발명의 구성 및 작용

본 발명은 상기의 목적을 달성하기 위하여 멀티채널 패킷 제어기 드라이버와 실시간 프로토콜(RTP) 블럭은 주기적으로 발생되는 하나의 타이머 인터럽트에 의해 동기화되는 하나의 응용 프로그램 태스크내에

SAMSUNG EX1002, Page 510 of 1031

구현하는 장치 및 방법을 제공함을 특징으로 한다.

또한, 본 발명은 상기의 목적을 달성하기 위하여 멀티채널 패킷 제어기로부터 실시간 프로토콜(RTP) 블럭으로 전달되는 음성 패킷들에 대해 멀티채널 패킷 제어기가 각 채널에 대한 음성 패킷을 제어하기 위해 시스템 초기화시 미리 고정적으로 메모리 영역이 할당된 버퍼들을 제어하며 실시간 프로토콜(RTP) 블럭으로부터 멀티 채널 패킷 제어기로 전달되는 음성 패킷들에 대해 실시간 프로토콜 블럭이 각 채널에 대한 음성 패킷을 제어하기 위해 시스템 초기화시 미리 고정적으로 메모리 영역이 할당된 버퍼들을 제어하도록 구성함을 특징으로 한다.

상기에서 미리 고정적으로 메모리 영역 할당시 각 단위 버퍼에 대해 음성 패킷뿐만이 아니라 12바이트의 고정적인 RTP 헤더도 미리 할당하도록 구성함을 특징으로 한다.

이하, 본 발명을 도면에 의거 상세히 설명하면 다음과 같다.

본 발명의 실시를 위한 시스템은 도3, 도5 및 도6 의 예시도에 도시한 바와 같이, 인터넷망과의 음성 패킷 데이터 송수신을 수행하는 UDP 블럭(205)과, 공중 전화망과의 음성 신호 송수신을 수행하면서 패킷화 또는 패킷 분리를 수행하는 멀티 채널 패킷 제어기(204)와, 상기 UDP 블럭(205) 및 멀티 채널 패킷 제어기(204)에서의 음성 패킷 데이터를 저장하는 메모리 블럭(210a)(210b)과, 인터럽트 발생때마다 상기 UDP 블럭(205)과 멀티 채널 패킷 제어기(204)간의 음성 데이터 전송을 제어하는 응용 프로그램 태스크 블럭(203)과, 주기적으로 인터럽트를 발생시키는 타이머(201)와, 이 타이머(201)에서 주기적으로 인터럽트 발생시 상기 응용 프로그램 태스크 블럭(203)으로 제공하는 인터럽트 서비스 블럭(202)으로 구성한다.

상기 응용 프로그램 태스크 블럭(203)은 실시간 프로토콜(RTP)과 멀티 채널 패킷 제어기 구동을 위한 회로를 포함하여 구성한다.

상기 메모리 블럭(210a)(210b)은 시스템 초기화시 각 채널의 음성 패킷에 대응하는 메모리 영역이 고정적으로 미리 할당되며 각 메모리 영역마다 RTP 헤더를 고정적으로 미리 할당하여 구성한다.

상기에서 메모리 블럭(210a)(210b)의 메모리 영역에 대응하여 음성 패킷의 저장 여부를 표시하는 플래그 및 음성 패킷 데이터의 저장 시작 위치를 나타내는 고정 어드레스를 구비한 수신 버퍼 디스크립터(220)와 전송 버퍼 디스크립터(230)를 더 구비하여 구성한다.

상기에서 메모리 블럭(210a)(210b)에 미리 할당된 메모리 영역은 링 방식으로 제어하는 것을 특징으로 한다.

이와같이 구성한 본 발명의 실시예에 대한 동작 및 작용 효과를 설명하면 다음과 같다.

먼저, 태스크 스케쥴링(Task Scheduling) 과정을 설명하면 다음과 같다.

응용 프로그램 태스크 블럭(203)은 주기적으로 반복되는 패킷 데이터의 특성을 고려하여 UDP 블럭(205)와 멀티채널 패킷 제어기(204)로부터 송수신되는 데이터들을 일괄적으로 하나의 태스크에서 처리한다.

이 경우 멀티채널 패킷 제어기(204)로부터의 인터럽트 이벤트가 발생할 때마다 데이터를 처리하는 방식을 사용하지 않고 주기적으로 폴링하는 방식을 사용한다.

즉, 트래픽 소스로부터의 인터럽트 이벤트 구동 방식의 경우 매 채널, 매 패킷 각각에 대해 인터럽트 서비스 스위칭 시간이나 태스크 스위칭 시간의 오버헤드가 적용된다.

예를 들어, 패킷화 주기가 '30msec'라 하고 하고 채널 개수가 'N'개라고 하면 공중 전화망에서는 수신 방향에 대해서 평균적으로 '30msec'마다 'N'번의 인터럽트 서비스 스위칭과 태스크 스위칭이 일어나야 한다.

또한, 반대 방향의 이벤트들에 대해서도 상기와 유사한 처리를 함으로 '30msec'라는 짧은 시간내에 수많은 이벤트들로 인한 스위칭 오버헤드가 적용되어 프로세스 효율이 훨씬 저하된다.

따라서, 본 발명은 도3 과 같은 태스크 스케쥴링 방식을 적용하면 주기적인 인터럽트 타이머(201)에 의해 '30msec'마다 한 번만 인터럽트가 발생하고 인터럽트 서비스 루틴(202)에서는 실시간 프로토콜(RTP)/멀티채널 패킷 제어기 구동을 동시에 실행할 수 있는 응용 프로그램 태스크 블럭(203)으로 이벤트 플래그를 한 번 전달하기만 하면 된다.

이러한 동작은 도4 의 신호 흐름과 동일한 과정으로 이루어진다.

일시 정지되어 있는 응용프로그램 태스크 블럭(203)은 타이머(201)에서 한 번의 타이머 인터럽트가 발생될 때마다 재실행된다.

그리고, 메모리 블럭(210a)(210b)은 시스템 초기화시 멀티 채널 패킷 제어기(204)로부터의 음성 패킷과 UDP 블럭(205)로부터의 음성 패킷을 저장하기 위한 메모리 영역이 미리 할당되고 그 각 메모리 영역에는 RTP 헤더도 미리 할당된다.

따라서, 멀티 채널 패킷 제어기(204)는 공중 전화망으로부터 전송되어 디지털 변환된 음성 데이터를 메모리 블럭(210a)의 해당 영역에 연속적으로 저장하며 UDP 블럭(205)은 인터넷 망으로부터 전송된 디지털 음성 데이터를 메모리 블럭(210b)의 해당 영역에 연속적으로 저장한다.

이때, 메모리 블럭(210a)의 해당 메모리 영역에 데이터가 모두 차면 멀티 채널 패킷 제어기(204)이 수신 버퍼 디스크립터(220)의 해당 영역의 상태 비트를 '1'로 갱신하며 메모리 블럭(210b)의 해당 메모리 영역에 데이터가 모두 차면 UDP 블럭(205)이 전송 버퍼 디스크립터(230)의 해당 영역의 상태 비트를 '1'로 갱신한다.

SAMSUNG EX1002, Page 511 of 1031

이 후, 타이머(201)에서 주기적인 인터럽트가 발생하면 인터럽트 서비스 블럭(202)은 주기적인 인터럽트의 발생을 알리는 이벤트 플래그를 응용 프로그램 태스크 블럭(203)에 제공한다.

이때, 응용 프로그램 태스크 블럭(203)은 현재 액티브되어 있는 음성 통화 채널에 대해 각각 멀티 채널 패킷 제어기(204)가 갱신하는 수신 버퍼 디스크립터(RxBD)(220)의 상태 비트를 점검한다.

이에 따라, 현재 액티브된 채널에 대응하는 수신 버퍼 디스크립터(220)의 해당 영역의 상태 비트가 '1'로서 공중 전화망으로부터 수신된 데이터가 존재하면 응용 프로그램 태스크 블럭(203)은 RTP 헤더에 프로토콜 데이터의 갱신을 수행한 후 멀티 채널 패킷 제어기(204)에 의해 채워진 RTP 데이터와 함께 UDP 블럭(205)으로 전송한 후 상기 상태 비트를 '0'을 갱신한다.

그리고, 응용 프로그램 태스크 블럭(203)은 해당 채널에 대응하는 UDP 블럭(205)의 포트를 점검하여 인터넷망으로부터 수신된 데이터가 존재하는지 판단한다.

이때, UDP 블럭(205)에서의 음성 패킷이 존재하면 응용 프로그램 태스크 블럭(203)은 RTP 헤더를 처리한 후 음성 패킷을 메모리 블럭(210b)의 해당 영역에 복사하고 송신 버퍼 디스크립터(TxBD)(230)의 해당 영역의 상태 비트를 '1'로 갱신한다.

이에 따라, 멀티 채널 패킷 제어기(204)는 메모리 블럭(210b)의 해당 영역에 저장된 RTP 데이터를 읽어 오디오 코덱으로 전송하고 송신 버퍼 디스크립터(230)의 해당 영역의 상태 비트를 '0'으로 갱신한다.

상기와 같은 동작을 'N'개의 채널 각각에 대하여 반복적으로 수행하며 이러한 동작이 모두 종료되면 응용 프로그램 태스크 블럭(203)은 바로 일시 정지 상태로 돌입한다.

즉, 본 발명에서의 버퍼 제어 과정을 도5 및 도6 을 참조하여 설명하면 다음과 같다.

메모리 블럭(210a)(210b)는 하나의 메모리 회로에 포함되며 각 음성 채널당 버퍼링 구조를 가지기 위해 시스템 초기화시 RTP 헤더와 음성 패킷을 저장하는 영역이 연속적으로 고정 할당한다.

도5 는 공중 전화망으로부터의 음성 패킷을 저장하여 전송하는 경우를 도시한 것으로, 멀티채널 패킷 제어기(204)는 각 오디오 코덱으로부터 음성 데이터를 패킷화하여 데이터를 메모리 블럭(210a)의 해당 버퍼 영역에 RTP 데이터(음성 패킷)을 복사하고 수신 버퍼 디스크립터(220)의 해당 영역의 상태 비트를 업데이터한다.

이에 따라, 응용 프로그램 태스크 블럭(203)은 주기적인 인터럽트 발생시 수신 버퍼 디스크립터(220)의 영역중 현재 액티브된 채널에 대응하는 영역의 상태 비트가 '1'인 경우 RTP 헤더와 RTP 데이터를 포함하는 UDP 데이터를 UDP 블럭(205)로 전송하고 그 RTP 헤더를 갱신한다.

또한, 도6 은 인터넷망으로부터의 음성 패킷을 저장하여 전송하는 경우를 도시한 것으로, 응용 프로그램 태스크 블럭(203)은 주기적인 인터럽트 발생시 UDP 블럭(205)를 점검하여 수신 데이터가 있는 경우 메모리 블럭(210b)의 영역중 현재 액티브된 채널에 대응하는 영역의 RTP 헤더 처리하고 수신된 음성 패킷을 복사한 후 송신 버퍼 디스크립터(230)의 해당 영역의 상태 비트를 갱신한다.

이에 따라, 멀티 채널 패킷 제어기(204)는 송신 버퍼 디스크립터(230)의 해당 영역의 상태 비트가 '1'인지를 확인하여 '1'인 경우 메모리 블럭(210b)의 해당 영역의 음성 패킷을 오디오 코덱으로 전송하고 상기 송신 버퍼 디스크립터(230)의 해당 영역의 상태 비트를 '0'으로 갱신한다.

상기 동작은 주기적인 인터럽트 발생시마다 동시에 수행한다.

발명의 효과

상기에서 상세히 설명한 바와 같이 본 발명은 공중전화망으로부터 수신되는 음성 패킷과 인터넷망으로부터 수신되는 음성 패킷을 중계하는 응용 프로그램 태스크에 의해 마이크로 프로세서에 가해지는 부하(load)를 최소화하여 동작 속도를 개선함으로써 보다 많은 채널을 수용할 수 있는 효과가 있다.

또한, 본 발명은 공중전화망 이외에도 종합정보통신망(ISDN)등이나 프레임 릴레이등 TDM 방식으로 음성을 전송하는 망과 인터넷망과 같은 패킷망간의 게이트웨이 시스템에도 적용하여 동일한 효과를 얻을 수 있는 효과가 있다.

(57) 청구의 범위

청구항 1

인터넷망과 공중 전화망간의 음성 데이터 송수신 방법에 있어서, 시스템 초기화시 소정 단위로 메모리 영역을 미리 고정 할당하고 그 각각의 영역에 RTP 헤더를 미리 고정 할당하는 제1 단계와, 복수개 채널에 대한 인터넷망과 공중 전화망으로부터의 음성 데이터를 상기에서 미리 할당된 해당 메모리 영역에 각기 연속으로 저장하는 제2 단계와, 주기적으로 하나의 인터럽트가 발생하는지 판단하는 제3 단계와, 상기에서 주기적인 하나의 인터럽트가 발생할 때마다 상기에서 저장된 복수개의 채널에 대한 인터넷망과 공중 전화망에서의 음성 패킷 데이터를 순차적으로 상호 전송하는 제4 단계를 수행함을 특징으로 하는 멀티 채널 음성 데이터 제어 방법.

청구항 2

공중 전화망과 인터넷망간의 음성 데이터를 저장하여 상호 전송하는 방법에 있어서, 시스템 초기화시 소정량 단위로 메모리 영역을 미리 할당하고 그 각각의 영역에 RTP 헤더를 미리 할당하는 제1 단계와, 태스크 일시 정지 상태에서 주기적인 인터럽트 발생을 판단하는 제2 단계와, 복수의 채널을 순차적으로 검색하여 액티브되는 채널을 판단하는 제3 단계와, 상기에서 액티브되는 채널을 판단하면 해당 채널의 데

SAMSUNG EX1002, Page 512 of 1031

이터가 존재하는지 판단하는 제4 단계와, 상기에서 해당 채널의 데이터가 존재하면 공중 전화망에서의 음성 데이터는 RTP 헤더와 함께 인터넷망으로 전송하고 동시에 인터넷망에서의 음성 데이터는 RTP 헤더를 처리하여 음성 데이터만을 공중 전화망으로 전송하는 제5 단계와, 상기에서 복수의 채널에 대한 음성 데이터 전송이 종료되면 태스크를 다시 일시 정지시키는 제6 단계를 수행하는 것을 특징으로 하는 멀티 채널 음성 데이터 제어 방법.

청구항 3

인터넷망과 공중 전화망간의 음성 데이터를 송수신하는 게이트웨이 시스템에 있어서, 인터넷망과의 음성 데이터 송수신을 수행하는 UDP 블럭과, 공중 전화망과의 음성 신호 송수신을 수행하는 멀티 채널 패킷 제어기와, 복수 채널에 대한 상기 UDP 블럭 및 멀티 채널 패킷 제어기에서의 음성 데이터를 해당 영역에 저장하는 메모리 블럭과, 주기적인 인터럽트 발생때마다 상기 메모리 블럭을 제어하여 복수 채널에 대한 상기 UDP 블럭과 멀티 채널 패킷 제어기간의 음성 데이터 전송을 실행시키는 응용 프로그램 태스크 블럭과, 주기적으로 인터럽트를 발생시켜 상기 실시간 프로토콜에 제공하는 타이머로 구성함을 특징으로 하는 멀티 채널 음성 데이터 제어 장치.

청구항 4

제3항에 있어서, 메모리 블럭은 시스템 초기화시 각 채널의 음성 패킷에 대응하는 영역을 고정적으로 미리 할당하여 구성함을 특징으로 하는 멀티 채널 음성 데이터 제어 장치.

청구항 5

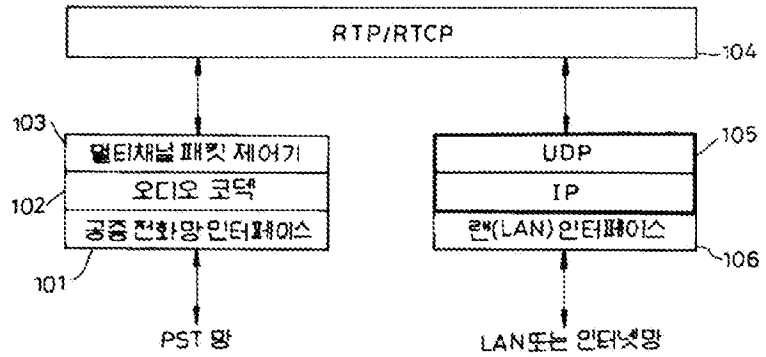
제4항에 있어서, 메모리 블럭은 시스템 초기화시 고정적으로 미리 할당되는 각 영역마다 RTP(Real-Time Protocol) 헤더를 고정적으로 미리 할당하여 구성함을 특징으로 하는 멀티 채널 음성 데이터 제어 장치.

청구항 6

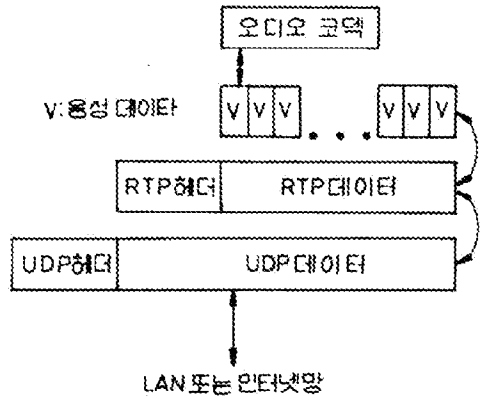
제3항에 있어서, 메모리 블럭은 미리 할당된 각 영역에 대응하여 음성 패킷의 저장 여부를 표시하는 상태 비트 및 음성 패킷 데이터의 저장 시작 위치를 나타내는 어드레스를 저장하는 송,수신 버퍼 디스크립터를 더 포함하여 구성함을 특징으로 하는 멀티 채널 음성 데이터 제어 장치.

도면

도면1

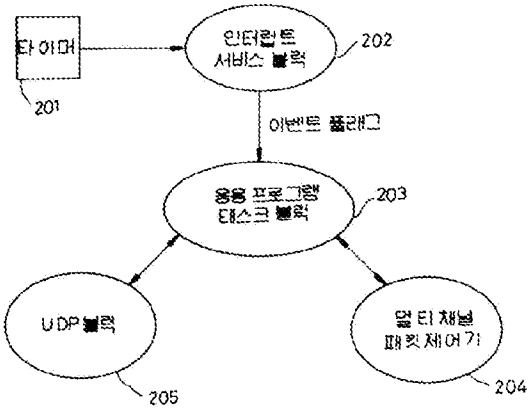


도면2

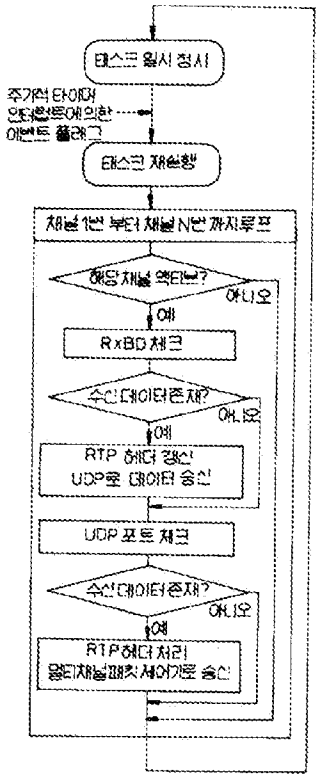


SAMSUNG EX1002, Page 513 of 1031

도면3

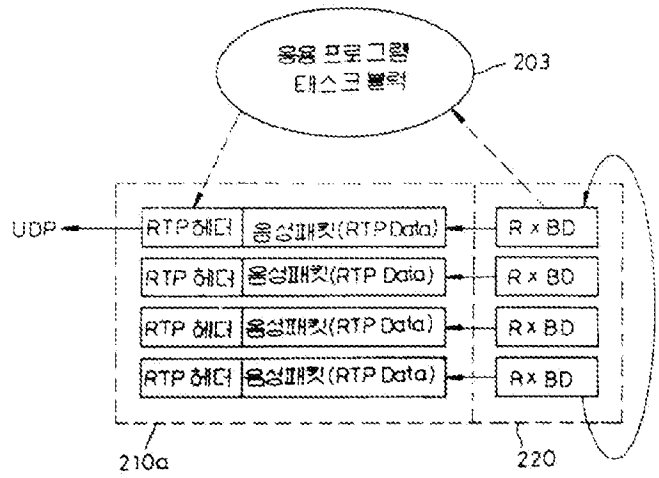


도면4

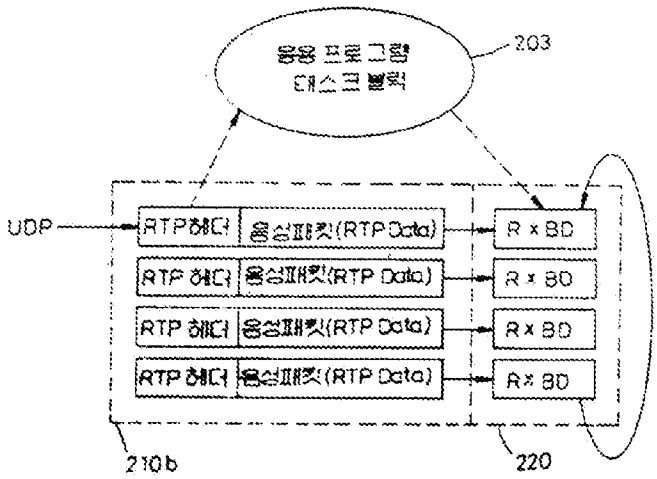


SAMSUNG EX1002, Page 514 of 1031

도면5



도면6



SAMSUNG EX1002, Page 515 of 1031

INFORMATION DISCLOSURE STATEMENT

First Inventor: Kuzmin, Andrey V.	**Conf. No.:** 3894
App. No.: 17/931,506	**Filed:** September 12, 2022
Examiner: Bataille, Pierre Miche	**Art Unit:** 2136
Title: STORAGE DEVICE WITH SUBDIVISIONS, SUBDIVISION QUERY, AND WRITE OPERATIONS	

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

References and Listing

Pursuant to 37 CFR 1.56, an Information Disclosure Statement listing references is provided herewith. Copies of any listed foreign and non-patent literature references are being submitted. Any foreign references may also include English abstract(s) and/or machine translation(s), but no representation is made as to their accuracy.

Pursuant to 37 CFR 1.97(g) and (h), Applicant makes no representation that the information is considered to be material to patentability. Additionally, inclusion on this list is not an admission that any of the cited documents are prior art in this application. Further, Applicant makes no representation regarding the completeness of this list, or that better art does not exist.

If the Examiner would like additional information regarding these references or if anything is unclear, the Examiner is invited to contact the undersigned for assistance.

No Disclaimers

To the extent that anything in the Information Disclosure Statement or the listed references could be construed as a disclaimer of any subject matter supported by the present application, Applicant hereby rescinds and retracts such disclaimer.

Other Applications of Assignee

Applicant wishes to draw the Examiner's attention to, and encourages the Examiner to review, the following co-owned patents and/or applications and their existing and ongoing prosecution history:

Application No.	Filing Date	Patent No.
13/767,723	2/14/2013	9,652,376
14/047,193	10/7/2013	9,229,854
14/951,708	11/25/2015	9,519,578
15/346,641	11/8/2016	9,710,377
15/621,888	6/13/2017	10,445,229
16/570,922	9/13/2019	11,118,457
16/751,925	1/24/2020	11,314,636
16/783,083	2/5/2020	11,334,479
16/794,805	2/19/2020	
16/833,547	3/28/2020	11,216,365
17/563,481	12/28/2021	11,487,657
17/563,467	12/28/2021	11,487,656
17/571,394	1/7/2022	
17/877,881	7/29/2022	
17/931,516	9/12/2022	
17/952,964	9/23/2022	
17/952,230	9/24/2022	

SAMSUNG EX1002, Page 517 of 1031

Applicant notes that cited references, office actions, responses, and notices of allowance currently exist or will exist with reference to the above-referenced matters. Applicant also understands that the Examiner has access to sophisticated Patent Office computing systems that provide ready access to the full file histories of these matters including, for example, specifications, drawings, pending claims, cited art, office actions, responses, declarations, and notices of allowance. Rather than submit copies of these file histories, Applicant respectfully requests that the Examiner continue to review these file histories online for past, current, and future information about these matters that may be relevant to examination of the present application. With respect to parent applications of the present application, Applicant understands that, in accordance with MPEP 609, the Examiner of the present application will consider information which has been considered by the Patent Office in the parent applications. If the Examiner cannot readily access these file histories and information, Applicant would be pleased to provide any portion of any of the file histories or information at any time upon specific Examiner request.

Timing of Disclosure

This Information Disclosure Statement is being filed after receipt of a First Office Action, but before the mailing date of a Final Action and before the mailing date of a Notice of Allowance.

This Statement is accompanied by the fees set forth in 37 CFR 1.17(p). The Commissioner is hereby authorized to charge any additional fees which may be required or to credit any overpayment to Account No. 11-1410.

Respectfully submitted,

KNOBBE, MARTENS, OLSON & BEAR, LLP

Dated: December 13, 2022

By: /Aaron M. Davis/
Aaron M. Davis
Registration No. 62,027
Registered Practitioner
(858) 707-4000

56774878

SAMSUNG EX1002, Page 518 of 1031

INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Application No.	17/931,506
	Filing Date	September 12, 2022
	First Named Inventor	Kuzmin, Andrey V.
	Art Unit	2136
(Multiple sheets used when necessary)	Examiner	Bataille, Pierre Miche
SHEET 1 OF 2	Attorney Docket No.	RMMS.003C12

U.S. PATENT DOCUMENTS					
Examiner Initials	Cite No.	Document Number *Number - Kind Code (if known)* Example: 1,234,567 B1	Publication Date MM-DD-YYYY	Name	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear
	1	5,963,977	10-05-1999	Gold	
	2	6,892,287	05-10-2005	Millard	
	3	7,631,138	12-08-2009	Gonzalez	
	4	8,341,339	12-25-2012	Boyle	
	5	11,307,995	04-19-2022	Lercari	
	6	11,487,656	11-01-2022	Kuzmin et al.	
	7	11,487,657	11-01-2022	Kuzmin et al.	
	8	2006/0004957	01-05-2006	Hand III	
	9	2011/0041039	02-17-2011	Harari	
	10	2011/0238892	09-29-2011	Tsai	
	11	2011/0238943	09-29-2011	Devendran et al.	
	12	2011/0264843	10-27-2011	Haines	
	13	2012/0155492	06-21-2012	Abel	
	14	2012/0198128	08-02-2012	Van Aken	
	15	2012/0303875	11-29-2012	Benhase	
	16	2013/0097236	04-18-2013	Khorashadi	
	17	2013/0166824	06-27-2013	Shim	

FOREIGN PATENT DOCUMENTS						
Examiner Initials	Cite No.	Foreign Patent Document *Country Code-Number-Kind Code* Example: JP 1234567 A1	Publication Date MM-DD-YYYY	Name	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear	T[1]
	18	JP 2001051889	02-23-2001	Sharp Corp.		X
	19	JP 2011123863	06-23-2011	Phison Electronics Corp.		X
	20	JP 2011203916	10-13-2011	Toshiba Corp.		X
	21	KR 20000025612	05-06-2000	LG Electronics Inc.		X
	22	RD 527033	03-10-2008			

Examiner Signature	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 next communication to applicant.

T[1] - Place a check mark in this area when an English language Translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Application No.	17/931,506
	Filing Date	September 12, 2022
	First Named Inventor	Kuzmin, Andrey V.
	Art Unit	2136
(Multiple sheets used when necessary)	Examiner	Bataille, Pierre Miche
SHEET 2 OF 2	Attorney Docket No.	RMMS.003C12

NON PATENT LITERATURE DOCUMENTS			
Examiner Initials	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	T[1]
	23	Y. H. Chang et al. "An efficient FTL design for multi-chipped solid-state drives," 2010 IEEE 16th intl. conference on embedded and real-time computing systems and applications, 2010, pp. 237-246.	
	24	Eric Seppanen, Matthew T. O'Keefe, David J. Lilja, Department of Electrical and Computer Engineering, University of Minnesota, ""High Performance Solid State Storage Under Linux,"" 4-10-2010, MSST '10 Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), Pages 1-12.	

56774681

Examiner Signature	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 next communication to applicant.

T[1] - Place a check mark in this area when an English language Translation is attached.

PTO/SB/06 (09-11)
Approved for use through 1/31/2014. 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	Application or Docket Number 17/931,506	Filing Date 09/12/2022	☐ To be Mailed

ENTITY: ☐ LARGE ☑ SMALL ☐ MICRO

APPLICATION AS FILED - PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA		RATE ($)	FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A		N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A		N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A		N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*		x $50 =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*		x $240 =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 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))					
* If the difference in column 1 is less than zero, enter "0" in column 2.				TOTAL	

APPLICATION AS AMENDED - PART II

		(Column 1)		(Column 2)	(Column 3)			
AMENDMENT	12/13/2022	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	* 30	Minus	** 30	= 0		x $50 =	0
	Independent (37 CFR 1.16(h))	* 3	Minus	*** 3	= 0		x $240 =	0
	☐ Application Size Fee (37 CFR 1.16(s))							
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	0

		(Column 1)		(Column 2)	(Column 3)			
AMENDMENT		CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=		x $0 =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=		x $0 =	
	☐ Application Size Fee (37 CFR 1.16(s))							
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.	LIE
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".	/KIM R WATSON/
*** 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.

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.
17/931,506	09/12/2022	Andrey V. Kuzmin	RMMS.003C12	3894

20995 7590 11/23/2022

KNOBBE MARTENS OLSON & BEAR LLP
2040 MAIN STREET
FOURTEENTH FLOOR
IRVINE, CA 92614

EXAMINER	
BATAILLE, PIERRE MICHE	

ART UNIT	PAPER NUMBER
2136	

NOTIFICATION DATE	DELIVERY MODE
11/23/2022	ELECTRONIC

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

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

efiling@knobbe.com
jayna.cartee@knobbe.com

Office Action Summary	Application No. 17/931,506	Applicant(s) Kuzmin et al.	
	Examiner PIERRE MICHEL BATAILLE	Art Unit 2136	AIA (FITF) Status Yes

-- 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 3 MONTHS 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 10/31/2022.

☐ A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on _____.

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

3)☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on _____; the restriction requirement and election have been incorporated into this action.

4)☐ 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*

5) ☐ Claim(s) 1-31 is/are pending in the application.

5a) Of the above claim(s) 1 is/are withdrawn from consideration.

6) ☐ Claim(s) _____ is/are allowed.

7) ☑ Claim(s) 2-31 is/are rejected.

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

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

* If any claims have been determined allowable, you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see http://www.uspto.gov/patents/init_events/pph/index.jsp or send an inquiry to **PPHfeedback@uspto.gov.**

Application Papers

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

11)☑ The drawing(s) filed on 09/12/2022 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).

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).

Certified copies:

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)).

** 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) ☑ Information Disclosure Statement(s) (PTO/SB/08a and/or PTO/SB/08b) Paper No(s)/Mail Date 10/31/2022.

3) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date _____.

4) ☐ Other: _____.

Notice of Pre-AIA or AIA Status

1. The present application, filed on or after March 16, 2013, is being examined under the first inventor to file provisions of the AIA.

- Claims 2-31 are now pending in the application under prosecution and have been examined. Claim 1 has been canceled.
- The specification has not been checked to the extent necessary to determine the presence of all possible minor errors.
- The specification should be amended to reflect the status of all related application, whether patented or abandoned. Therefore, applications noted by their serial number and/or attorney docket number should be updated with correct serial number and patent number if patented.
- The first instance of all acronyms or abbreviation should be spelled out for clarity, whether or not considered well known in the art.

In the response to this Office action, the Examiner respectfully requests that support be shown for language added to any original claims on amendment and any new claims. That is, indicate support for newly added claim language by specifically pointing to page(s) and line numbers in the specification and/or drawing figure(s). This will assist the Examiner in prosecuting this application.

Examiner cites particular columns and line numbers in the references as applied to the claims below for the convenience of the applicant. Although the specified citations are representative of the teachings in the art and are applied to the specific limitations within the

individual claim, other passages and figures may apply as well. It is respectfully requested that, in preparing responses, the applicant fully consider the references in entirety as potentially teaching all or part of the claimed invention, as well as the context of the passage as taught by the prior art or disclosed by the examiner.

37 C.F.R. § 1.83(a) requires the Drawings to illustrate or show all claimed features. Applicant must clearly point out the patentable novelty that they think the claims present, in view of the state of the art disclosed by the references cited or the objections made, and must also explain how the amendments avoid the references or objections. See 37 C.F.R. § 1.111(c).

Double Patenting

2. 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. A nonstatutory double patenting rejection is appropriate where the conflicting claims are not identical, but at least one examined application claim is not patentably distinct from the reference claim(s) because the examined application claim is either anticipated by, or would have been obvious over, the reference claim(s). See, e.g., *In re Berg*, 140 F.3d 1428, 46 USPQ2d 1226 (Fed. Cir. 1998); *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); *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) or 1.321(d) may be used to overcome an actual or provisional rejection based on nonstatutory double patenting provided the reference application or patent either is shown to be commonly owned with the examined application, or claims an invention made as a result of activities undertaken within the scope of a joint research agreement. See MPEP § 717.02 for applications subject to examination under the first inventor to file provisions of the AIA as explained in MPEP § 2159. See MPEP § 2146 *et seq.* for applications not subject to examination under the first inventor to file provisions of the AIA. A terminal disclaimer must be signed in compliance with 37 CFR 1.321(b).

The USPTO Internet website contains terminal disclaimer forms which may be used. Please visit www.uspto.gov/patent/patents-forms. The filing date of the application in which the form is filed determines what form (e.g., PTO/SB/25, PTO/SB/26, PTO/AIA/25, or PTO/AIA/26) should be used. A web-based eTerminal Disclaimer may be filled out completely online using web-screens. An eTerminal Disclaimer that meets all requirements is auto-processed and approved immediately upon submission. For more information about eTerminal Disclaimers, refer to www.uspto.gov/patents/process/file/efs/guidance/eTD-info-I.jsp.

3. Claims 2-31 are provisionally rejected on the ground of nonstatutory double patenting as being unpatentable over claims 2-25 of copending Application No. 17/931,516. Although the claims at issue are not identical, they are not patentably distinct from each other because :

"the storage device comprising: flash memory divided into zones, each zone of the zones being a subdivision comprising a set of erase units, and each erase unit of the set of erase units comprising host accessible pages" of Application 17/931,506; and

"the storage device comprising: flash memory divided into zones, each zone of the zones being a subdivision comprising a set of erase units, and each erase unit of the set of erase units comprising host accessible pages" of application 17/931,516

achieve common host configuration and management of tables used to perform multi-plane or multi-die space assignment, reads, writes and associated management function.

This is a provisional nonstatutory double patenting rejection.

Claim 2 (17/931506)	Claim 2 (17/931,516)
A storage device comprising: flash memory divided into zones, each zone of the zones being a subdivision comprising a set of erase units, and each erase unit of the set of erase units comprising host accessible pages, wherein each zone: is associated with a logical address range, has a zone size for which the storage	A host comprising: at least one processor; and non-transitory machine readable media storing instructions that, when executed by the at least one processor cause operations to be performed, the operations comprising: sending a query command to a storage device, wherein the storage device comprises flash memory divided into zones, each zone of the zones being a subdivision comprising one or more erase units, and each erase unit of the one or more erase units comprising host accessible pages, wherein each zone: is associated with a logical address range, has a zone size for which the storage

device is operable to receive a management command from a host that results in the storage device performing an erasure of one or more erase units of the set of erase units, is in a full state when there are no host accessible pages available for writing without first erasing, and is in an empty state when all host accessible pages are available for writing without first erasing; and	device is operable to receive a management command from the host that results in the storage device performing an erasure of the one or more erase units of the zone, is in a full state when there are no host accessible pages available for writing without first erasing, and is in an empty state when all host accessible pages are available for writing without first erasing;
logic operable to: return information about the zones in response to a query command from the host, the information about the zones comprising a list of the zones accessible by the host, a zone state, the zone size, logical addresses associated with respective zones, and	processing information about the zones, the information about the zones provided from the storage device in response to the query command, and the information about the zones comprising a list of the zones accessible by the host, a zone state, the zone size, logical addresses associated with respective zones, and
information representing an extent to which the respective zones include host accessible pages available for writing without first erasing;	information representing an extent to which the respective zones include host accessible pages available for writing without first erasing; and
concurrently write data to the set of erase units of a specified zone using a multi-plane write command; and update the information representing the extent to which the specified zone includes host accessible pages available for writing without first erasing, wherein the logic comprises at least one of hardware logic or instructions stored on non-transitory machine-	issuing a write command accompanied by a write address to the storage device to cause a write to the one or more erase units of a specified zone of the zones, wherein the write address is dependent upon the information about the zones provided from the storage device.

readable media that, when executed, control operation of hardware circuitry.	

Claim 11 (17/931,506)	Claim 11 (17/931,516)
A storage device comprising: flash memory divided into zones, each zone of the zones being a subdivision comprising a set of erase units, and each erase unit of the set of erase units comprising host accessible pages, wherein each zone: is associated with a logical address range, has a zone size associated with a management command from a host, transitions to a full state in response to no longer having host accessible pages available for writing without first erasing, and in response to the storage device receiving the management command associated therewith transitions to an empty state such that all host accessible pages are available for writing without first erasing; and logic operable to: return information about zones stored by the storage device in response to	A storage system comprising: flash memory divided into zones, each zone of the zones being a subdivision comprising one or more erase units, and each erase unit of the one or more erase units comprising host accessible pages, wherein each zone: is associated with a logical address range, has a zone size associated with a management command that results in an erasure of the one or more erase units of the zone, is in a full state when there are no host accessible pages available for writing without first erasing, and is in an empty state when all host accessible pages are available for writing without first erasing; a memory controller in communication with the flash memory, the memory controller

a query command from the host, the information about the zones comprising: a list of the zones accessible by the host, a respective logical address associated with a zone of the zones stored, the zone size, information indicating, for respective zones in which there are no host accessible pages available for writing without first erasing the respective zone, that the respective zones are in the full state, and information representing, for respective zones, an extent to which the respective zones include host accessible pages available for writing without first erasing; concurrently write data to the set of erase units of a specified zone using a common page address; update the information representing the extent to which the specified zone includes host accessible pages available for writing without first erasing; and when the specified zone has no host accessible pages available for writing without first erasing the zone after concurrently writing data to the set of erase units of the specified zone, update the information indicating that the specified zone is in the full state, wherein the logic comprises at	comprising storage arranged to retain information about the zones, the information about the zones comprising a list of the zones accessible by the host, a zone state, the zone size, logical addresses associated with respective zones, and information representing an extent to which the respective zones include host accessible pages available for writing without first erasing; and a host operable to issue the management command, send a query command to the memory controller, process the information about the zones that is received from the memory controller in response to the query command, and issue a write command accompanied by a write address to the memory controller to cause a write to one or more erase units of a specified zone of the zones, wherein the write address is dependent upon the information about the zones.

least one of hardware logic or instructions stored on non-transitory machine-readable media that, when executed, control operation of hardware circuitry.	

Claim 20 reciting the method corresponding to the storage system of claim 11 is therefore rejected in view of claim 11 of Application 17/931,506.

Conclusion

4. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Related prior art include:

WO 2009084724 A1 (ASANO et al) teaching NAND type flash memory which, when data is stored therein, once erases data in units of block and thereafter to perform writing, a type which performs reading and writing in units called a page.

US 20140075097 (NAGAI) teaching semiconductor storage device and method for creating partitions capable of writing into all the addresses or partitions not capable of it per partition.

US 20110296094 teaching host computer requesting storage of some logical block addresses more than once with existing pages containing duplicated data are marked as obsolete, and the new data is written to the next available erase block.

US 20110283046 (KOSEKI) teaching flash controller configured to control an operation of the flash memory, wherein a storage controller, before accessing the flash memory, queries the flash controller as to whether the flash memory is accessible.

US 20090172286 (LASSER et al) teaching method to include determining an available capacity in a cache storage portion of a self-caching storage device, determining a ratio of cache flushing steps to host write commands if the available capacity is below a desired threshold and interleaving cache flushing steps with host write commands to achieve the ratio.

US 20080219057 (LI) teaching non-volatile memory and methods includes cached page copying using a minimum number of data latches for each memory cell, an adaptive full-sequence programming performed concurrently with the processing.

US 20100325351 (BENNETT) teaching process of recovering space that is no longer being used for storage of current data that may interfere with the rapid access to data in other memory locations of the memory system cannot accept further writes without being reclaimed (erased).

US 20110289255 (WANG et al) teaching processing circuit to write data and associated logical addresses into available physical pages of other data blocks of particular data writing groups without erasing these data blocks in advance.

D. M. Han, "Fast erase algorithm using flash translation layer in NAND-type flash memory," in IEEE Transactions on Consumer Electronics, vol. 57, no. 4, pp. 1749-1755, November 2011.

Contact Information

5. Any inquiry concerning this communication or earlier communications from the examiner should be directed to PIERRE MICHEL BATAILLE whose telephone number is (571)272-4178. The examiner can normally be reached Monday - Thursday 7-6 ET.

Examiner interviews are available via telephone, in-person, and video conferencing using a USPTO supplied web-based collaboration tool. To schedule an interview, applicant is encouraged to use the USPTO Automated Interview Request (AIR) at http://www.uspto.gov/interviewpractice.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Charles Rones can be reached on (571) 272-4085. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of published or unpublished applications may be obtained from Patent Center. Unpublished application information in Patent Center is available to registered users. To file and manage patent submissions in Patent Center, visit: https://patentcenter.uspto.gov. Visit https://www.uspto.gov/patents/apply/patent-center for more information about Patent Center and https://www.uspto.gov/patents/docx for information about filing in DOCX format. For additional questions, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/PIERRE MICHEL BATAILLE/
Primary Examiner, Art Unit 2136