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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
10 April 2014 (10.04.2014)

WIPO | PCT

(10) International Publication Number
WO 2014/055445 A1

(51) International Patent Classification:
G06F 12/00 (2006.01)

(21) International Application Number:
PCT/US2013/062723

(22) International Filing Date:
30 September 2013 (30.09.2013)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
13/645,822 5 October 2012 (05.10.2012) US

(71) Applicants: **WESTERN DIGITAL TECHNOLOGIES, INC.** [US/US]; 3355 Michelson Drive, Suite 100, Irvine, CA 92612 (US). **SKYERA, INC.** [US/US]; 1704 Automation Parkway, San Jose, CA 95131 (US).

(72) Inventors: **TOMLIN, Andrew, J.**; c/o Western Digital Technologies, Inc., 3355 Michelson Drive, Suite 100, Irvine, CA 92612 (US). **JONES, Justin**; c/o Western Digital Technologies, Inc., 3355 Michelson Drive, Suite 100, Irvine, CA 92612 (US). **DANILAK, Radoslav**; 10425 Glencore Drive, Cupertino, CA 95014 (US). **MULLENDORE, Rodney, N.**; 5813 Killarney Circle, San Jose, CA 95138 (US).

(74) Agent: **DELANEY, Karoline, A.**; Knobbe Martens Olson & Bear, LLP, 2040 Main Street, 14th Floor, Irvine, CA 92614 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, 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, 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, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, 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, KM, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: METHODS, DEVICES AND SYSTEMS FOR PHYSICAL-TO-LOGICAL MAPPING IN SOLID STATE DRIVES

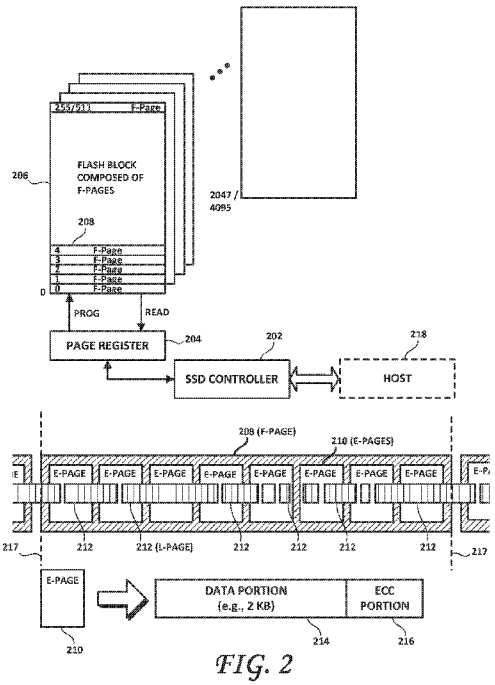


FIG. 2

(57) Abstract: A data storage device comprises a plurality of non-volatile memory devices storing physical pages, each stored at a predetermined physical location. A controller may be coupled to the memory devices and configured to access data stored in a plurality of logical pages (L-Pages), each associated with an L-Page number that enables the controller to logically reference data stored in the physical pages. A volatile memory may comprise a logical-to-physical address translation map that enables the controller to determine a physical location, within the physical pages, of data stored in each L~Page. The controller may be configured to maintain, in the memory devices, journals defining physical-to-logical correspondences, each journal covering a predetermined range of physical pages and comprising a plurality of entries that associate one or more physical pages to each L-Page. The controller may read the journals upon startup and rebuild the address translation map from the read journals.

WO 2014/055445 A1

Declarations under Rule 4.17:

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

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

Published:

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

METHODS, DEVICES AND SYSTEMS FOR PHYSICAL-TO-LOGICAL MAPPING IN SOLID STATE DRIVES

BACKGROUND

[0001] Due to the nature of flash memory in solid state drives (SSDs), data is typically programmed by pages and erased by blocks. A page in an SSD is typically 8-16 kilobytes (KB) in size and a block consists of a large number of pages (e.g., 256 or 512). Thus, a particular physical location in an SSD (e.g., a page) cannot be directly overwritten without overwriting data in pages within the same block, as is possible in a magnetic hard disk drive. As such, address indirection is needed. Conventional data storage device controllers, which manage the flash memory on data storage devices such as SSDs and interface with the host system, use a Logical to Physical (L2P) mapping system known as Logical Block Addressing (LBA) that is part of the flash translation layer (FTL). When new data comes in replacing older data already written, the data storage device controller causes the new data to be written in a new location and update the logical mapping to point to the new physical location. Since the old physical location no longer holds valid data, it will eventually need to be erased before it can be written again.

[0002] Conventionally, a large L2P map table maps logical entries to physical address locations on an SSD. This large L2P map table, which may reside in a volatile memory such as dynamic random access memory (DRAM), is usually updated as writes come in, and saved to non-volatile memory in small sections. For example, if random writing occurs, although the system may have to update only one entry, it may nonetheless have to save to the non-volatile memory the entire table or a portion thereof, including entries that have not been updated, which is inherently inefficient.

[0003] Fig. 1 shows aspects of a conventional Logical Block Addressing (LBA) scheme for an SSD. As shown therein, a map table 104 contains one entry for every logical block 102 defined for the data storage device's flash memory 106. For example, a 64 GB SSD that supports 512 byte logical blocks may present itself to the host as having 125,000,000 logical blocks. One entry in the map table 104 contains the current location of each of the 125,000,000 logical blocks in the flash memory 106. In a

conventional SSD, a flash page holds an integer number of logical blocks (i.e., a logical block does not span across flash pages). In this conventional example, an 8 KB flash page would hold 16 logical blocks (of size 512 bytes). Therefore, each entry in the logical-to-physical map table 104 contains a field 108 identifying the flash die on which the logical block is stored, a field 110 identifying the flash block on which the logical block is stored, another field 112 identifying the flash page within the flash block and a field 114 identifying the offset within the flash page that identifies where the logical block data begins in the identified flash page. The large size of the map table 104 prevents the table from being held inside the SSD controller. Conventionally, the large map table 104 is held in an external DRAM connected to the SSD controller. As the map table 104 is stored in volatile DRAM, it must be restored when the SSD powers up, which can take a long time, due to the large size of the table.

[0004] When a logical block is read, the corresponding entry in the map table 104 is read to determine the location in flash memory to be read. A read is then performed to the flash page specified in the corresponding entry in the map table 104. When the read data is available for the flash page, the data at the offset specified by the map entry is transferred from the SSD to the host. When a logical block is written, the corresponding entry in the map table 104 is updated to reflect the new location of the logical block. It is to be noted that when a logical block is written, the flash memory will initially contain at least two versions of the logical block; namely, the valid, most recently written version (pointed to by the map table 104) and at least one other, older version thereof that is stale and is no longer pointed to by any entry in the map table 104. These “stale” data are referred to as garbage, which occupies space that must be accounted for, collected, erased and made available for future use.

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] Fig. 1 shows aspects of a conventional Logical Block Addressing (LBA) scheme for SSDs.

[0006] Fig. 2 is a diagram showing aspects of the physical and logical data organization of a data storage device according to one embodiment.

[0007] Fig. 3 shows a logical-to-physical address translation map and illustrative

entries thereof, according to one embodiment.

[0008] Fig. 4 shows aspects of a method for updating a logical-to-physical address translation map and for creating an S-Journal entry, according to one embodiment.

[0009] Fig. 5 is a block diagram of an S-Journal, according to one embodiment.

[0010] Fig. 6 shows an exemplary organization of one entry of an S-Journal, according to one embodiment.

[0011] Fig. 7 is a block diagram of a superblock (S-Block), according to one embodiment.

[0012] Fig. 8 shows another view of a super page (S-Page), according to one embodiment.

[0013] Fig. 9A shows relationships among the logical-to-physical address translation map, S-Journals and S-Blocks, according to one embodiment.

[0014] Fig. 9B is a block diagram of an S-Journal Map, according to one embodiment.

[0015] Fig. 10 is a block diagram illustrating aspects of a method of updating the logical-to-physical address translation map, according to one embodiment.

[0016] Fig. 11 is a block diagram illustrating further aspects of a method of updating the logical-to-physical address translation map, according to one embodiment.

[0017] Fig. 12 is a block diagram illustrating still further aspects of a method of updating the logical-to-physical address translation map, according to one embodiment.

[0018] Fig. 13 is a block diagram illustrating yet further aspects of a method of updating the logical-to-physical address translation map, according to one embodiment.

[0019] Fig. 14 is a block diagram illustrating aspects of a method of updating S-Journals and the S-Journal Map, according to one embodiment.

[0020] Fig. 15 is a block diagram illustrating further aspects of a method of updating S-Journals and the S-Journal Map, according to one embodiment.

[0021] Fig. 16 is a block diagram illustrating still further aspects of a method of updating S-Journals and the S-Journal Map, according to one embodiment.

[0022] Fig. 17 is a block diagram illustrating yet further aspects of a method of updating S-Journals and the S-Journal Map, according to one embodiment.

[0023] Fig. 18 is a block diagram illustrating aspects of garbage collection, according

to one embodiment.

[0024] Fig. 19 is a block diagram illustrating further aspects of garbage collection, according to one embodiment.

[0025] Fig. 20 is a block diagram illustrating still further aspects of garbage collection, according to one embodiment.

[0026] Fig. 21 is a block diagram illustrating yet further aspects of garbage collection, according to one embodiment.

[0027] Fig. 22 is a block diagram illustrating aspects of garbage collecting a system block, according to one embodiment.

[0028] Fig. 23 is a block diagram illustrating further aspects of garbage collecting a system block, according to one embodiment.

[0029] Fig. 24 is a block diagram illustrating still further aspects of garbage collecting a system block, according to one embodiment.

[0030] Fig. 25 is a block diagram illustrating other aspects of garbage collecting a system block, according to one embodiment.

[0031] Fig. 26 is a block diagram illustrating still other aspects of garbage collecting a system block, according to one embodiment.

DETAILED DESCRIPTION

System Overview

[0032] Fig. 2 is a diagram showing aspects of the physical and logical data organization of a data storage device according to one embodiment. In one embodiment, the data storage device is an SSD. In another embodiment, the data storage device is a hybrid drive including flash memory and rotating magnetic storage media. The disclosure is applicable to both SSD and hybrid implementations, but for the sake of simplicity the various embodiments are described with reference to SSD-based implementations. A data storage device controller 202 according to one embodiment may be configured to be coupled to a host, as shown at reference numeral 218. The host 218 may utilize a logical block addressing (LBA) scheme. While the LBA size is normally fixed, the host can vary the size of the LBA dynamically. For example, the LBA size may vary by interface and interface mode. Indeed, while 512 bytes is

most common, 4 KB is also becoming more common, as are 512+ (520, 528 etc.) and 4 KB+ (4 KB+8, 4K+16 etc.) formats. As shown therein, the data storage device controller 202 may comprise or be coupled to a page register 204. The page register 204 may be configured to enable the controller 202 to read data from and store data to the data storage device. The controller 202 may be configured to program and read data from an array of flash memory devices responsive to data access commands from the host 218. While the description herein refers to flash memory generally, it is understood that the array of memory devices may comprise one or more of various types of non-volatile memory devices such as flash integrated circuits, Chalcogenide RAM (C-RAM), Phase Change Memory (PC-RAM or PRAM), Programmable Metallization Cell RAM (PMC-RAM or PMCm), Ovonic Unified Memory (OUM), Resistance RAM (RRAM), NAND memory (e.g., single-level cell (SLC) memory, multi-level cell (MLC) memory, or any combination thereof), NOR memory, EEPROM, Ferroelectric Memory (FeRAM), Magnetoresistive RAM (MRAM), other discrete NVM (non-volatile memory) chips, or any combination thereof.

[0033] The page register 204 may be configured to enable the controller 202 to read data from and store data to the array. According to one embodiment, the array of flash memory devices may comprise a plurality of non-volatile memory devices in die (e.g., 128 dies), each of which comprises a plurality of blocks, such as shown at 206 in Fig. 2. Other page registers 204 (not shown), may be coupled to blocks on other die. A combination of flash blocks, grouped together, may be called a Superblock or S-Block. In some embodiments, the individual blocks that form an S-Block may be chosen from one or more dies, planes or other levels of granularity. An S-Block, therefore, may comprise a plurality of flash blocks, spread across one or more die, that are combined together. In this manner, the S-Block may form a unit on which the Flash Management System (FMS) operates. In some embodiments, the individual blocks that form an S-Block may be chosen according to a different granularity than at the die level, such as the case when the memory devices include dies that are sub-divided into structures such as planes (i.e., blocks may be taken from individual planes). According to one embodiment, allocation, erasure and garbage collection may be carried out at the S-Block level. In other embodiments, the FMS may perform data operations according to

other logical groupings such as pages, blocks, planes, dies, etc.

[0034] In turn, each of the flash blocks 206 comprises a plurality of flash pages (F-Pages) 208. Each F-Page may be of a fixed size such as, for example, 16 KB. The F-Page, according to one embodiment, is the size of the minimum unit of program for a given flash device. As also shown in Fig. 2, each F-Page 208 may be configured to accommodate a plurality of physical pages, hereinafter referred to as E-Pages 210. The term "E-Page" refers to a data structure stored in flash memory on which an error correcting code (ECC) has been applied. According to one embodiment, the E-Page 210 may form the basis for physical addressing within the data storage device and may constitute the minimum unit of flash read data transfer. The E-Page 210, therefore, may be (but need not be) of a predetermined fixed size (such as 2 KB, for example) and determine the size of the payload (e.g., host data) of the ECC system. According to one embodiment, each F-Page 208 may be configured to fit a predetermined plurality of E-Pages 210 within its boundaries. For example, given 16 KB size F-Pages 208 and a fixed size of 2 KB per E-Page 210, eight E-Pages 210 fit within a single F-Page 208, as shown in Fig. 2. In any event, according to one embodiment, a power of 2 multiple of E-Pages 210, including ECC, may be configured to fit into an F-Page 208. Each E-Page 210 may comprise a data portion 214 and, depending on where the E-Page 210 is located, may also comprise an ECC portion 216. Neither the data portion 214 nor the ECC portion 216 need be fixed in size. The address of an E-Page uniquely identifies the location of the E-Page within the flash memory. For example, the E-Page's address may specify the flash channel, a particular die within the identified flash channel, a particular block within the die, a particular F-Page and, finally, the E-Page within the identified F-Page.

[0035] To bridge between physical addressing on the data storage device and logical block addressing by the host, a logical page (L-Page) construct is introduced. An L-Page, denoted in Fig. 2 at reference numeral 212 may comprise the minimum unit of address translation used by the FMS. Each L-Page, according to one embodiment, may be associated with an L-Page number. The L-Page numbers of L-Pages 212, therefore, may be configured to enable the controller 202 to logically reference host data stored in one or more of the physical pages, such as the E-Pages 210. The L-

Page 212 may also be utilized as the basic unit of compression. According to one embodiment, unlike F-Pages 208 and E-Pages 210, L-Pages 212 are not fixed in size and may vary in size, due to variability in the compression of data to be stored. Since the compressibility of data varies, a 4 KB amount of data of one type may be compressed into a 2 KB L-Page while a 4 KB amount of data of a different type may be compressed into a 1 KB L-Page, for example. Due to such compression, therefore, the size of L-Pages may vary within a range defined by a minimum compressed size of, for example, 24 bytes to a maximum uncompressed size of, for example, 4 KB or 4 KB+. Other sizes and ranges may be implemented. As shown in Fig. 2, L-Pages 212 need not be aligned with the boundaries of E-Page 210. Indeed, L-Pages 212 may be configured to have a starting address that is aligned with an F-Page 208 and/or E-Page 210 boundary, but also may be configured to be unaligned with either of the boundaries of an F-Page 208 or E-Page 210. That is, an L-Page starting address may be located at a non-zero offset from either the start or ending addresses of the F-Pages 208 or the start or ending addresses of the E-Pages 210, as shown in Fig. 2. As the L-Pages 212 are not fixed in size and may be smaller than the fixed-size E-Pages 210, more than one L-Page 212 may fit within a single E-Page 210. Similarly, as the L-Pages 212 may be larger in size than the E-Pages 210, L-Pages 212 may span more than one E-Page, and may even cross the boundaries of F-Pages 210, shown in Fig. 2 at numeral 217.

[0036] For example, where the LBA size is 512 or 512+ bytes, a maximum of, for example, eight sequential LBAs may be packed into a 4 KB L-Page 212, given that an uncompressed L-Page 212 may be 4 KB to 4 KB+. It is to be noted that, according to one embodiment, the exact logical size of an L-Page 212 is unimportant as, after compression, the physical size may span from few bytes at minimum size to thousands of bytes at full size. For example, for 4 TB SSD device, 30 bits of addressing may be used to address each L-Page 212 to cover for an amount of L-Pages that could potentially be present in such a SSD.

[0037] Fig. 3 shows a logical-to-physical address translation map and illustrative entries thereof, according to one embodiment. As the host data is referenced by the host in L-Pages 212 and as the data storage device stores the L-Pages 212 in one or more contiguous E-Pages 210, a logical-to-physical address translation map is required

to enable the controller 202 to associate an L-Page number of an L-Page 212 to one or more E-Pages 210. Such a logical-to-physical address translation map is shown in Fig. 3 at 302 and, in one embodiment, is a linear array having one entry per L-Page 212. Such a logical-to-physical address translation map 302 may be stored in a volatile memory, such as a DRAM or SRAM. Fig. 3 also shows the entries in the logical-to-physical address translation map for four different L-Pages 212, which L-Pages 212 in Fig. 3 are associated with L-Page numbers denoted as L-Page 1, L-Page 2, L-Page 3 and L-Page 4. According to one embodiment, each L-Page stored in the data storage device may be pointed to by a single and unique entry in the logical-to-physical address translation map 302. Accordingly, in the example being developed herewith, four entries are shown. As shown at 302, each entry in the map 302 may comprise an L-Page number, which may comprise an identification of the physical page (e.g., E-Page) containing the start address of the L-Page being referenced, the offset of the start address within the physical page (e.g., E-Page) and the length of the L-Page. In addition, a plurality of ECC bits may provide error correction functionality for the map entry. For example, and as shown in Fig. 3, and assuming an E-Page size of 2 KB, L-Page 1 may be referenced in the logical-to-physical address translation map 302 as follows: E-Page 1003, offset 800, length 1624, followed by a predetermined number of ECC bits (not shown). That is, in physical address terms, the start of L-Page 1 is within (not aligned with) E-Page 1003, and is located at an offset from the starting physical location of the E-Page 1003 that is equal to 800 bytes. Compressed L-Page 1, furthermore, extends 1,624 bytes, thereby crossing an E-Page boundary to E-Page 1004. Therefore, E-Pages 1003 and 1004 each store a portion of the L-Page 212 denoted by L-Page number L-Page 1. Similarly, the compressed L-Page referenced by L-Page number L-Page 2 is stored entirely within E-Page 1004, and begins at an offset therein of 400 bytes and extends only 696 bytes within E-Page 1004. The compressed L-Page associated with L-Page number L-Page 3 starts within E-Page 1004 at an offset of 1,120 bytes (just 24 bytes away from the boundary of L-Page 2) and extends 4,096 bytes past E-Page 1005 and into E-Page 1006. Therefore, the L-Page associated with L-Page number L-Page 3 spans a portion of E-Page 1004, all of E-Page 1005 and a portion of E-Page 1006. Finally, the L-Page associated with L-Page number L-Page 4

begins within E-Page 1006 at an offset of 1,144 bytes, and extends 3,128 bytes to fully span E-Page 1007, crossing an F-Page boundary into E-Page 1008 of the next F-Page.

[0038] Collectively, each of these constituent identifier fields (E-Page, offset, length and ECC) making up each entry of the logical-to-physical address translation map 302 may be, for example, 8 bytes in size. That is, for an exemplary 4 TB drive, the address of the E-Page may be 32 bits in size, the offset may be 12 bits (for E-Page data portions up to 4 KB) in size, the length may be 10 bits in size and the ECC field may be provided. Other organizations and bit-widths are possible. Such an 8 byte entry may be created each time an L-Page is written or modified, to enable the controller 202 to keep track of the host data, written in L-Pages, within the flash storage. This 8-byte entry in the logical-to-physical address translation map may be indexed by an L-Page number of LPN. In other words, according to one embodiment, the L-Page number functions as an index into the logical-to-physical address translation map 302. It is to be noted that, in the case of a 4 KB sector size, the LBA is the same as the LPN. The LPN, therefore, may constitute the address of the entry within the volatile memory. When the controller 202 receives a read command from the host 218, the LPN may be derived from the supplied LBA and used to index into the logical-to-physical address translation map 302 to extract the location of the data to be read in the flash memory. When the controller 202 receives a write command from the host, the LPN may be constructed from the LBA and the logical-to-physical address translation map 302 may be modified. For example, a new entry therein may be created. Depending upon the size of the volatile memory storing the logical-to-physical address translation map 302, the LPN may be stored in a single entry or broken into, for example, a first entry identifying the E-Page containing the starting address of the L-Page in question (plus ECC bits) and a second entry identifying the offset and length (plus ECC bits). According to one embodiment, therefore, these two entries may together correspond and point to a single L-Page within the flash memory. In other embodiments, the specific format of the logical-to-physical address translation map entries may be different from the examples shown above.

S-Journals and S-Journal Map

[0039] As the logical-to-physical address translation map 302 may be stored in a

volatile memory, it may need to be rebuilt upon startup or any other loss of power to the volatile memory. This, therefore, requires some mechanism and information to be stored in a non-volatile memory that will enable the controller 202 to reconstruct the logical-to-physical address translation map 302 before the controller can "know" where the L-Pages are stored in the non-volatile memory after startup or after a power-fail event. According to one embodiment, such mechanism and information are embodied in a construct that may be called a System Journal, or S-Journal. According to one embodiment, the controller 202 may be configured to maintain, in the plurality of non-volatile memory devices (e.g., in one or more of the blocks 206 in one or more die, channel or plane), a plurality of S-Journals defining physical-to-logical address correspondences. According to one embodiment, each S-Journal covers a pre-determined range of physical pages (e.g., E-Pages). According to one embodiment, each S-Journal may comprise a plurality of journal entries, with each entry being configured to associate one or more physical pages, such as E-Pages, to the L-Page number of each L-Page. According to one embodiment, each time the controller 202 restarts or whenever the logical-to-physical address translation map 302 is to be rebuilt either partially or entirely, the controller 202 reads the S-Journals and, from the information read from the S-Journal entries, rebuilds the logical-to-physical address translation map 302.

[0040] Fig. 4 shows aspects of a method for updating a logical-to-physical address translation map and for creating an S-Journal entry, according to one embodiment. As shown therein, to ensure that the logical-to-physical address translation map 302 is kept up-to-date, whenever an L-Page is written or otherwise updated as shown at block B41, the logical-to-physical address translation map 302 may be updated as shown at B42. As shown at B43, an S-Journal entry may also be created, storing therein information pointing to the location of the updated L-Page. In this manner, both the logical-to-physical address translation map 302 and the S-Journals are updated when new writes occur (e.g., as the host issues writes to non-volatile memory, as garbage collection/wear leveling occurs, etc.). Write operations to the non-volatile memory devices to maintain a power-safe copy of address translation data may be configured, therefore, to be triggered by newly created journal entries (which may be just a few

bytes in size) instead of re-saving all or a portion of the logical-to-physical address translation map, such that Write Amplification (WA) is reduced. The updating of the S-Journals ensures that the controller 202 can access a newly updated L-Page and that the logical-to-physical address translation map 302 may be reconstructed upon restart or other information-erasing power event affecting the volatile memory in which the logical-to-physical address translation map is stored. Moreover, in addition to their utility in rebuilding the logical-to-physical address translation map 302, the S-Journals are useful in enabling effective Garbage Collection (GC). Indeed, the S-Journals may contain the last-in-time update to all L-Page numbers, and also may contain stale entries, entries that do not point to a valid L-Page.

[0041] According to one embodiment, the S-Journals may be the main flash management data written to the non-volatile memory. S-Journals may contain mapping information for a given S-Block and may contain the Physical-to-Logical (P2L) information for a given S-Block. Fig. 5 is a block diagram showing aspects of an S-Journal, according to one embodiment. As shown therein, each S-Journal 502 covers a predetermined physical region of the non-volatile memory such as, for example, 32 E-Pages as shown at 506, which are addressable using 5 bits. Each S-Journal 502 may be identified by an S-Journal Number, which may be part of a header 504 that could include other information about the S-Journal. The S-Journal Number may comprise a portion of the address of the first physical page covered by the S-Journal. For example, the S-Journal Number of S-Journal 502 may comprise, for example, the 27 Most Significant Bits (MSb) of the first E-Page address covered by this S-Journal 502.

[0042] Fig. 6 shows an exemplary organization of one entry 602 of an S-Journal 502, according to one embodiment. Each entry 602 of the S-Journal 502 may point to the starting address of one L-Page, which is physically addressed in E-Pages. Each entry 602 may comprise, for example, a number (5, for example) of Least Significant Bits (LSbs) of the address of the E-Page containing the start L-Page. The full E-Page address is obtained by concatenating these 5 LSbs with the 27 MSbs of the S-Journal Number in the header 504. In addition, the entry 602 may comprise the L-Page number, its offset within the identified E-Page and its size. For example, each entry 602 of an S-Journal may comprise the 5 LSbs of the address of first E-Page covered by this

S-Journal entry, 30 bits of L-Page number, 9 bits of E-Page offset and 10 bits of L-Page size, adding up to an overall size of about 7 bytes. Various other internal journal entry formats may be used in other embodiments.

[0043] According to one embodiment, due to the variability in the compression or the host configuration of the data stored in L-Pages, a variable number of L-Pages may be stored in a physical area, such as a physical area equal to 32 E-Pages, as shown at 506. As a result of the use of compression and the consequent variability in the sizes of L-Pages, S-Journals may comprise a variable number of entries. For example, according to one embodiment, at maximum compression, an L-Page may be 24 bytes in size and an S-Journal may comprise over 2,500 entries, referencing an equal number of L-Pages, one L-Page per S-Journal entry 602.

[0044] As noted above, an S-Journal may be configured to contain mapping information for a given S-Block. More precisely, according to one embodiment, S-Journals contain the mapping information for a predetermined range of E-Pages within a given S-Block. Fig. 7 is a block diagram of a S-Block, according to one embodiment. As shown therein, an S-Block 702 may comprise one flash block (F-Block) 704 (as also shown at 206 in Fig. 2) per die. An S-Block, therefore, may be thought of as a collection of F-Blocks, one F-Block per die, that are combined together to form a unit of the Flash Management System. According to one embodiment, allocation, erasure and GC may be managed at the S-Block level. Each F-Block 704, as shown in Fig. 7, may comprise a plurality of flash pages (F-Page) such as, for example, 256 or 512 F-Pages. An F-Page, according to one embodiment, may be the size of the minimum unit of program for a given non-volatile memory device. Fig. 8 shows a super page (S-Page), according to one embodiment. As shown therein, an S-Page 802 may comprise one F-Page per F-Block of an S-Block, meaning that an S-Page spans across an entire S-Block.

Relationships Among Various Data Structures

[0045] Fig. 9A shows relationships among the logical-to-physical address translation map, the S-Journal map and S-Blocks, according to one embodiment. Reference 902 denotes an entry in the logical-to-physical address translation map (stored in DRAM in one embodiment). According to one embodiment, the logical-to-physical address translation map may be indexed by L-Page number, in that there may be one entry 902

per L-Page in the logical-to-physical address translation map. The physical address of the start of the L-Page in the flash memory and the size thereof may be given in the map entry 902; namely by E-Page address, offset within the E-Page and the size of the L-Page. As noted earlier, the L-Page, depending upon its size, may span one or more E-Pages and may span F-Pages and F-Blocks as well.

[0046] As shown at 904, the volatile memory (e.g., DRAM) may also store a System Journal (S-Journal) map. An entry 904 in the S-Journal map stores information related to where an S-Journal is physically located in the non-volatile memory. For example, the 27 MSbs of the E-Page physical address where the start of the L-Page is stored may constitute the S-Journal Number (as previously shown in Fig. 5). The S-Journal map entry 904 in the volatile memory may also include the address of the S-Journal in non-volatile memory, referenced in system E-Pages. From the S-Journal map entry 904 in volatile memory, System S-Block Information 908 may be extracted. The System S-Block Information 908 may be indexed by System S-Block (S-Block in the System Band) and may comprise, among other information regarding the S-Block, the size of any free or used space in the System S-Block. Also from the S-Journal map entry 904, the physical location (expressed in terms of E-Pages in the System Band) of the referenced S-Journal in non-volatile memory 910 may be extracted.

[0047] The System Band, according to one embodiment, does not contain L-Page data and may contain File Management System (FMS) meta-data and information. The System Band may be configured as lower page only for reliability and power fail simplification. During normal operation, the System Band need not be read except during Garbage Collection. The System Band may be provided with significantly higher overprovisioning than the data band for overall WA optimization. Other bands include the Hot Band, which may contain L-Page data and is frequently updated, and the Cold Band, which may be less frequently updated and may comprise more static data, such as data that may have been collected as a result of GC. According to one embodiment, the System, Hot and Cold Bands may be allocated by an S-Block basis.

[0048] As noted above, each of these S-Journals in non-volatile memory may comprise a collection of S-Journal entries and cover, for example, 32 E-Pages worth of data. These S-Journals in non-volatile memory 910 enable the controller 202 to rebuild

not only the logical-to-physical address translation map in volatile memory, but also the S-Journal map, the User S-Block Information 906, and the System S-Block Information 908, in volatile memory.

[0049] Fig. 9B is a block diagram of an S-Journal Map 912, according to one embodiment. The S-Journal Map 912 may be indexed by S-Block number and each entry thereof may point to the start of the first S-Journal for that S-Block which, in turn, may cover a predetermined number of E-Pages (e.g., 32) of that S-Block. The controller 202 may be further configured to build or rebuild a map of the S-Journals and store the resulting S-Journal Map in volatile memory. That is, upon restart or upon the occurrence of another event in which power fails or after a restart subsequent to error recovery, the controller 202 may read the plurality of S-Journals in a predetermined sequential order, build a map of the S-Journals stored in the non-volatile memory devices based upon the sequentially read plurality of S-Journals, and store the built S-Journal Map 912 in the volatile memory.

Updating the logical-to-physical Address Translation Map

[0050] Figs. 10-13 are block diagrams illustrating aspects of a method of updating the logical-to-physical address translation map, according to one embodiment. As shown in Fig. 10, a logical-to-physical address translation map 1002 contains an entry (e.g., a location of an L-Page) for L-Page 100, which has a length of 3,012 bytes. In this example, L-Page 100 is stored in S-Block 15, as shown at 1006. A buffer 1004 (such as a static random access memory (SRAM)) in or coupled to the controller 202 may store the S-Journal that contains the P2L information for S-Block 15 in which the L-Page 100 resides. What is shown in the buffer 1004 may actually reside in DRAM in some embodiments. The User S-Block Info 906, whose entries are indexed by S-Block, may comprise for each S-Block, among other information regarding the S-Block, the (e.g., exact or approximate) size of the free or used space in the S-Block. As shown in Fig. 10, the entry in the User S-Block Info 906 for S-Block 15 is shown. Fig. 10 shows an illustrative state of these constituent functional blocks before an update to L-Page 100 is processed by the controller 202.

[0051] As shown in Fig. 11 at 1102, an updated L-Page 100 is received, with a new length of 1,534 bytes. Responsive to the receipt of the updated L-Page 100, the L-

Page 100 information may be fetched from the logical-to-physical address translation map 1002 and the length (3,012 bytes) of the (now obsolete) L-Page 100 may be extracted therefrom and used to correspondingly and precisely increase the tracked free space value of S-Block 15. In particular, the User S-Block Information 906 may be updated with data indicating that S-Block 15 now has 3,012 additional bytes of free space, now that the original data for L-Page 100 is now stale (e.g., obsolete).

[0052] As shown in Fig. 12, the logical-to-physical address translation map 1002 may be updated to accommodate the updated L-Page. For example, the length information is now 1,534 bytes. Thereafter, the S-Journal in the buffer 1004 for the particular portion of S-Block 15 where the update occurs may now be updated with the new L-Page information, including length information and the L-Page newly received from, for example, a compressor may be read into the buffer 1004. While the new L-Page is in the buffer, the entry for L-Page 100 in the logical-to-physical address translation map 1002 may temporarily reflect its location in the buffer, as shown by the arrow labeled "1". Later, the updated L-Page 100 may be flushed to the Hot S-Block 1008, at the E-Page address specified by the newly-created entry in the now updated S-Journal still in the buffer 1004. The mapping table entry in the logical-to-physical address translation map 1002 is updated to reflect the physical E-Page address and offset of the L-Page's final destination, as suggested by arrow "2".

[0053] As shown in Fig. 13, at some later point in time such as, for example, after accumulating a sufficient number of new entries, the S-Journal in the volatile memory buffer 1004 may be written out to non-volatile memory, such as to the System S-Block 1010. The System S-Block 1010 may be a portion of the flash memory allocated by the controller firmware to store S-Journals. The saving of the S-Journal in non-volatile memory enables the later reconstruction of the logical-to-physical address translation map, as needed.

Updating the S-Journals and S-Journal Map

[0054] Figs. 14-17 are block diagrams illustrating aspects of a method of updating S-Journals and the S-Journal Map in the System Band, according to one embodiment. The figures illustrate additional details relating to the update mechanisms of S-Journal triggered by the same example update to L-Page 100. As shown therein, an S-Journal

Map 1402 in volatile memory (e.g., synchronous dynamic random-access memory (SDRAM)), may contain an entry that points to the physical location, in non-volatile memory, of the S-Journal that contains the P2L mapping data associated with L-Page 100. In the illustrative example, the relevant S-Journal (i.e., S-Journal for a particular portion of S-Block 15 storing L-Page 100) is located in System S-Block 3 (at 1408), starting at E-Page 42.

[0055] Thereafter, as shown Fig. 15, L-Page 100 is updated, with a length of 1,534 bytes. The information (e.g., E-Page address, offset and length) regarding current L-Page 100 may then be fetched from the logical-to-physical address translation map 1002. Thereafter, the address of the E-Page storing L-Page 100 may be used to extract the S-Journal Number used to locate the S-Journal entry within the S-Journal Map 1402. For example, in the case wherein, as shown at Figs. 5 and 6, the S-Journal Number may comprise the 27 MSbs of the first 32-bit E-Page address. This S-Journal number may then be used to index into the S-Journal Map 1402 to obtain the location, which may then be used to identify the System S-Block that contains the S-Journal of interest. Thereafter, as shown at Fig. 15, the System S-Block Information 908 may be updated to reflect the fact that the journal entry for L-Page 100 in the S-Journal at S-Block 3 is now invalid, as the space previously occupied by that entry has now been de-allocated, in view of the recent update to L-Page 100. That de-allocated space is now free space, and must be accounted for. Alternatively, used space may be accounted for and the free space derived therefrom. The entry in the System S-Block information 908 for S-Block 3 may, therefore, be incremented by a space taken by one S-Journal entry; namely, 7 bytes in the exemplary implementation being developed herein. This empty space may thereafter be taken into account during GC of the System Band.

[0056] As shown in Fig. 16, L-Page 100 may then be written to the buffer 1004, as shown at 1005, whereupon the logical-to-physical address translation map 1002 may be updated to point to the buffer 1004 (indicating the physical location where L-Page 100 is stored) and to store the length of the updated L-Page 100 (1,534 in the example being developed herein). It is to be noted that, at this point, System S-Block 3 still includes an S-Journal entry that points to S-Block 15 at reference 1006 as containing L-Page 100. However, the System S-Block Info for S-Block 3 has been updated so as to indicate that

the space presently occupied by the now outdated S-Journal entry within the System S-Block 3 at reference 1408 is, in fact, free space.

[0057] Fig. 16 also shows a different S-Journal in the buffer. In one embodiment, the S-Journal in the buffer 1004 is for recording new L-Pages written to the Open Hot S-Block 7. In this example, the S-Journal in the buffer 1004 (which has not yet been flushed to the non-volatile memory) is updated with the L-Page number of updated L-Page 100 (now in the buffer also) and its length information, as indicated by the arrow at 1007 in Fig. 16. L-Page 100 may now be flushed from the buffer 1004 to the currently open Hot S-Block 7, as referenced at 1404 in Fig. 16. At that point, the S-Journal in the buffer is updated to reflect that updated L-Page 100 has been written to the non-volatile memory. Alternatively, the logical-to-physical address translation map 1002 and the S-Journal may be updated with the address of the updated L-Page 100 prior to the updated L-Page 100 being written to the non-volatile memory. In one embodiment, an updated entry in that S-Journal will contain the E-Page pointed to by the new L-Page 100's entry in the logical-to-physical address translation map 1002 and additional information about L-Page 100, as previously shown in Fig. 6.

[0058] As shown in Fig. 17, after accumulating sufficient data, the S-Journal in the buffer 1004 may be written out to an open System S-Block 1 (referenced at 1406). In this example, this S-Journal is written out to E-Page 19 of open System S-Block 1. The S-Journal Map 1402 now comprises one entry indicating that a portion of S-Block 15 is stored in System S-Block 3, E-Page 42. However, the entry within that S-Journal which pointed to the old location of L-Page 100 within S-Block 15 is no longer valid. The S-Journal Map 1402 also comprises one entry indicating that S-Journal for S-Block 7 is stored in System S-Block 1, E-Page 19. That S-Journal includes a valid entry that points to open Hot S-Block 7 at 1404 as the location in non-volatile memory of the updated L-Page 100. In this manner, the S-Journals containing the P2L information are updated in the buffer 1004, and are eventually written out to non-volatile memory. In addition, the S-Journal Map 1402 may be suitably updated to point to such newly updated S-Journals. Moreover, the logical-to-physical address translation map 1002 may be suitably updated to point to the correct open Hot S-Block.

[0059] Advantageously, on a new write, the S-Journal construct as shown and

described herein minimizes write overhead. According to one embodiment, a new write operation necessitates writing a new S-Journal entry in the non-volatile memory. Since there is only a small amount of S-Journal entry data per L-Page generated (e.g., 7 bytes as described herein), the WA due to system data writes is reduced as compared to conventional systems. S-Journal data is effectively a massive command history. Updated S-Journal data is pooled up and written out sequentially to the System Band. The S-Journal system shown and described herein is efficient, as the mapping information that is generated is the same as the data to be written, with no additional unchanged entries, as only that which is changed is written to non-volatile (e.g., flash) memory, as opposed to all or a portion of a logical-to-physical map.

[0060] On power up, all the S-Journal data in the System Band may be read and processed into DRAM to rebuild the logical-to-physical address translation map in volatile memory. According to one embodiment, this is done in the order in which the S-Blocks are allocated to the System Band. In one embodiment, hardware support is used to enable this large data load to occur quickly to meet power-up timing requirement (this may not be practical without hardware, since this is essentially generating the logical-to-physical address translation map from the command history). According to one embodiment, an S-Journal Map is also constructed at power-up to point to valid S-Journals stored in the System Band.

Garbage Collection of User Data S-Blocks

[0061] Figs. 18-21 are block diagrams illustrating aspects of garbage collection, according to one embodiment. As shown therein, the data in the User S-Block Information 906 may be scanned to select the "best" S-Block to garbage collect. There are a number of criteria that may be evaluated to select which S-Block to garbage collect. For example, the best S-Block to garbage collect may be that S-Block having the largest amount of free space and the lowest Program Erase (PE) count. Alternatively, these and/or other criteria may be weighted to select the S-Block to be garbage collected. For purposes of example, the S-Block selected to be garbage collected in Figs. 18-21 is S-Block 15, which is referenced by information entry 15 within the User S-Block Information 906, showing some amount + 3,012 bytes of free space (the additional 3,012 bytes to account for the recently obsoleted L-Page 100). It is to be

noted that the User S-Block Information 906 may comprise, among other items of information, a running count of the number of PE cycles undergone by each tracked S-Block, which may be evaluated in deciding which S-Block to garbage collect. As shown at 1006, S-Block 15 has a mix of valid data (hashed blocks) and invalid data (non-hashed blocks).

[0062] Now that S-Block 15 has been selected for GC, the S-Journal Map (see 912 in Fig. 9B) may be consulted (e.g., indexed into by the S-Block number) to find the location in non-volatile memory (e.g., E-Page address) of the corresponding S-Journals for that S-Block. To illustrate an example, one S-Journal pointed to by the S-Journal Map 912 is located using the S-Journal Number (27 MSb of E-Page Address), and read into the buffer 1004, as shown in Fig. 18. That is, the 1 or more E-Pages in the System S-Block 1804 pointed to by S-Journal Map 912 are accessed and the S-Journal stored beginning at that location may be read into the buffer 1004. In one embodiment, the S-Journal map also contains the length of the S-Journal since an S-Journal may span one or more E-Pages. An S-Journal may be quite large and, therefore, may be read in pieces and processed as available.

[0063] Thereafter, each P2L entry in the S-Journal in the buffer 1004 may then be compared to the corresponding entry in the logical-to-physical address translation map 1802. For each entry in the S-Journal in the buffer 1004, it may be determined whether the physical address for the L-Page of that entry matches the physical address of the same L-Page in the corresponding entry in the logical-to-physical address translation map 1802. If the two match, that entry in the S-Journal is valid. Conversely, if the address for the L-Page in the S-Journal does not match the entry for that L-Page in the logical-to-physical address translation map, that entry in the S-Journal is not valid.

[0064] According to one embodiment, as valid entries are found in the S-Journal whose entries are being parsed and compared, the referenced L-Pages may be read out of S-Block 15 and written to the buffer 1004, as shown in Fig. 19. The same process may be used for other S-Journals covering S-Block 15 until the entire S-Block is processed. As also shown in Fig. 19 at reference 1006, S-Block 15 now contains only invalid data. This is because the data indicated as valid by entries in S-Journals for S-Block 15 have been preserved and will soon be moved to a new S-Block. In the

illustrated example, as the entries in the example S-Journal of S-Block 15 in System S-Block 1804 point to such invalid data, that S-Journal is shown as being hashed, indicating that it is now stale. The logical-to-physical address translation map 1802 may then be updated, generating a new E-Page starting address for the valid data read into the buffer 1004. It is to be noted that during the update of the logical-to-physical address translation map, the map may be rechecked for valid entries and may be locked during the map update process to guarantee atomicity. The valid data will also necessitate new S-Journal entries be generated in S-Journal 1005, which in one embodiment is for the Cold S-Block 1801.

[0065] In one embodiment, the valid data may then be written out to the Cold S-Block 1801 (the Hot S-Block being used for recently written host data, not garbage collected data), as shown at Fig. 20. At some later time (e.g., after a sufficient number of entries have been populated), S-Journal 1005 may be written out to the System S-Block 1804 in the System Band. S-Block 15 has now been garbage collected and User S-Block Info 906 now indicates that the entire S-Block 15 is free space. S-Block 15 may thereafter be erased, its PE count updated and made available for new writes. It is to be noted that an invalid S-Journal is still present in System S-Block 1804. The space in flash memory in the System Band occupied by this invalid S-Journal may be garbage collected at some later time.

Garbage Collection on System S-Blocks

[0066] Figs. 22-26 are block diagrams illustrating aspects of garbage collecting a system block, according to one embodiment. In the example being developed in Figs. 22-26, it is assumed that System S-Block 3, shown at referenced 2208 in Fig. 22, has been picked for garbage collection. Once a System S-Block has been picked, all of the E-pages (and by extension, all S-Journals contained therein) within the picked System S-Block may be read (sequentially or non-sequentially) into the buffer 1004.

[0067] As suggested in Fig. 23, the S-Journal numbers for one or more of the S-Journals read into the buffer 1004 may then be extracted from the headers of the S-Journals. Each such System S-Journal number may then be used to look up in the S-Journal Map 2202 to determine whether the corresponding S-Journal is still valid. According to one embodiment, invalid S-Journals are those S-Journals whose S-Journal

number is not matched by a corresponding entry in the S-Journal Map 2202, which has the most updated information on where S-Journals are physically stored. For example, if the entry in the S-Journal Map 2202 for S-Journal Number "12345" points to an E-Page within System S-Block 120, the copy of S-Journal "12345" in S-Block 3 (the S-Block being garbage collected) is obsolete. Likewise, if the S-Journal Map entry instead points to the E-Page from S-Block 3 where S-Journal "12345" currently resides, S-Journal "12345" is still valid.

[0068] In one embodiment, a valid S-Journal being garbage collected may include a mix of valid and invalid entries, thus individual checking of the entries is needed. As shown in Fig. 24, each entry in each valid S-Journal 2402 may then be matched with a corresponding entry in the logical-to-physical address translation map 1802 in memory. That is, the E-Page address of the L-Page referenced by each S-Journal entry may be compared with the E-Page address of the L-Page specified in the logical-to-physical address translation map 1802. If the two match, that S-Journal entry is valid. According to one embodiment, it may be necessary to compare the E-Page address and the offset within the E-Page in the S-Journal with the E-Page address and the offset within the E-page of the L-Page specified in the logical-to-physical address translation map 1802. Conversely, if the E-page address (or E-page address and offset) for the L-Page in the S-Journal does not match the E-Page address (or E-page address and offset) in the entry for that L-Page in the address translation map 1802, that entry in the S-Journal is not valid.

[0069] According to one embodiment, as valid entries are identified in the S-Journal 2402 (whose entries are being parsed and compared), they may be copied to a new version 2502 of the S-Journal 2402, as shown in Fig. 25. The S-Journal 2502, according to one embodiment, may have the same S-Journal number as that of the S-Journal 2402. In this manner, each S-Journal loaded into the buffer 1004 from the S-Block picked for GC may be first determined to be valid or invalid at the journal level, and then, if determined valid, compacted to contain only valid entries. According to one embodiment, invalid S-Journal entries are simply not copied to the new version of the S-Journal 2502. Accordingly, it is expected that the new version 2502 of the S-Journal will be smaller (i.e., comprise fewer entries) than the older version 2402 thereof, presuming

that the S-Journal 2402 had one or more obsolete entries. It is to be noted that if an S-Journal has all valid entries, the size of the new version thereof will remain the same.

[0070] As shown in Fig. 26, the S-Journal 2502 (the new version of the S-Journal 2402, which is now invalid or obsolete) may then be written to the current open System S-Block which, in Fig. 26, is Open System S-Block 1, reference numeral 2206. Thereafter, the S-Journal Map 2202 may be updated with the new location of the S-Journal 2502 in Open System S-Block 1. At the end of this process, the S-Journals in System S-Block 3 (2208) have been garbage collected and System S-Block 3 may then be erased and made available for future programming.

[0071] While certain embodiments of the disclosure have been described, these embodiments have been presented by way of example only, and are not intended to limit the scope of the disclosure. Indeed, the novel methods, devices and systems described herein may be embodied in a variety of other forms. Furthermore, various omissions, substitutions and changes in the form of the methods and systems described herein may be made without departing from the spirit of the disclosure. The accompanying claims and their equivalents are intended to cover such forms or modifications as would fall within the scope and spirit of the disclosure. For example, those skilled in the art will appreciate that in various embodiments, the actual physical and logical structures may differ from those shown in the figures. Depending on the embodiment, certain steps described in the example above may be removed, others may be added. Also, the features and attributes of the specific embodiments disclosed above may be combined in different ways to form additional embodiments, all of which fall within the scope of the present disclosure. Although the present disclosure provides certain preferred embodiments and applications, other embodiments that are apparent to those of ordinary skill in the art, including embodiments which do not provide all of the features and advantages set forth herein, are also within the scope of this disclosure. Accordingly, the scope of the present disclosure is intended to be defined only by reference to the appended claims.

CLAIMS:

1. A data storage device, comprising:

a plurality of non-volatile memory devices, each configured to store a plurality of physical pages, each of the plurality of physical pages being stored at a predetermined physical location within the plurality of non-volatile devices;

a controller coupled to the plurality of memory devices and configured to program data to and read data from the plurality of memory devices, the data being stored in a plurality of logical pages (L-Pages), each of the plurality of L-Pages being associated with an L-Page number that is configured to enable the controller to logically reference data stored in one or more of the physical pages; and

a volatile memory comprising a logical-to-physical address translation map configured to enable the controller to determine a physical location, within one or more physical pages, of the data stored in each L-Page;

wherein the controller is configured to maintain, in the plurality of non-volatile memory devices, a plurality of journals defining physical-to-logical correspondences, each of the plurality of journals being associated with a journal number, each journal covering a pre-determined range of physical pages and comprising a plurality of journal entries, each entry being configured to associate one or more physical pages to each L-Page, wherein the controller is configured to read the plurality of journals upon startup and rebuild the address translation map stored in the volatile memory from the read plurality of journals.

2. The data storage device of claim 1, wherein the controller is further configured to, upon an update to one of the plurality of L-Pages, create a new entry in one of the plurality of journals.

3. The data storage device of claim 2, wherein write operations to the non-volatile memory devices to maintain a power-safe copy of translation data are configured to be triggered by newly created journal entries instead of saving at least portions of the translation map, such that write amplification is reduced.

4. The data storage device of claim 2, wherein the new entry indicates a physical location, within a physical page, of a start of the updated L-Page.

5. The data storage device of claim 2, wherein the controller is further configured to update a free space accounting by an amount corresponding to a length of the L-Page prior to being updated.

6. The data storage device of claim 1, wherein at least one of the plurality of L-Pages is unaligned with physical page boundaries.

7. The data storage device of claim 1, wherein the physical pages are implemented as Error Correcting Code (ECC) pages (E-Pages) and wherein the plurality of devices comprises a plurality of flash memory blocks, each flash memory block comprising a plurality of flash memory pages (F-Pages), each of the F-Pages comprising a plurality of the E-Pages, each of the plurality of E-Pages being stored at a predetermined physical location within the plurality of devices.

8. The data storage device of claim 2, wherein the controller is further configured to update the translation map with the physical location, within one or more physical pages, of the data referenced by the L-Page number of the updated L-Page.

9. The data storage device of claim 1, further comprising a plurality of super blocks (S-Blocks), each comprising one or more flash memory blocks per device and wherein each of the plurality of journal entries is configured to associate one or more of the physical pages of the S-Block to each L-Page.

10. The data storage device of claim 9, wherein the controller is further configured to garbage collect by at least:

selecting one of the plurality of S-Blocks to garbage collect;

comparing each entry in a journal for the selected S-Block to entries in the

translation map and designating entries that match as valid and entries that do not match as invalid;

reading the L-Pages corresponding to the valid entries;

writing the read L-Pages to respective physical addresses within the plurality of non-volatile memory devices;

updating the translation map for the valid entries to point to the respective physical addresses; and

generating new journal entries for the entries for which the translation map was updated.

11. The data storage device of claim 9, wherein the controller is further configured to garbage collect by at least:

selecting one of the plurality of S-Blocks to garbage collect;

reading the physical pages of the selected S-Block;

comparing L-Page numbers in the read physical pages of the selected S-Block to entries in the translation map and designating entries that match as valid and entries that do not match as invalid;

writing the L-Pages corresponding to the valid entries to respective physical addresses within the plurality of non-volatile memory devices;

updating the translation map for the valid entries to point to the respective physical addresses; and

generating new journal entries for the entries for which the translation map was updated.

12. The data storage device of claim 10, wherein selecting comprises weighing free space and program erase (PE) count in determining which S-Block to select.

13. The data storage device of claim 1, wherein each journal number comprises a predetermined number of most significant bits of an address of a first physical page covered by the journal.

14. The data storage device of claim 1, wherein each of the plurality of journal entries comprises:

an L-Page number, and

a physical address location.

15. The data storage device of claim 1, wherein each of the plurality of journal entries comprises:

an L-Page number;

a physical address location of a physical page, and

an L-Page size.

16. The data storage device of claim 1, wherein each of the plurality of journal entries comprises:

a predetermined number of least significant bits of an address of a physical page that includes a start of an L-Page;

an address;

an L-Page size; and

an offset into the physical page.

17. The data storage device of claim 1, wherein the plurality of L-Pages are configured to be compressed and to vary in size, and wherein the plurality of journal numbers are configured to reference a greater number of L-Pages of smaller size or a lesser number of L-Pages of greater size.

18. The data storage device of claim 1, wherein the controller is further configured to read the plurality of journals upon startup in a predetermined sequential order and rebuild the translation map stored in volatile memory based upon the sequentially read plurality of journals.

19. The data storage device of claim 1, wherein the controller is further

configured to build a journal map based on the plurality of journals.

20. The data storage device of claim 1, wherein the controller is further configured to:

read the plurality of journals upon startup in a predetermined sequential order;

build a map of the journals stored in the non-volatile memory devices based upon the sequentially read plurality of journals, and

store the built map of the journals in the volatile memory.

21. The data storage of device of claim 1, wherein, among journal entries associated with a given L-Page, only a last-in-time updated journal entry associated with the given L-Page is valid.

22. The data storage device of claim 1, wherein the controller is further configured to maintain a system journal map of the plurality of journals in the volatile memory, each entry in the system journal map pointing to a location in the non-volatile memory devices where one of the plurality of journals is stored.

23. A method of controlling a data storage device comprising a volatile memory and a plurality of non-volatile memory devices, each of the plurality of non-volatile devices being configured to store a plurality of physical pages, each of the plurality of physical pages being stored at a predetermined physical location within the plurality of non-volatile devices, the method comprising:

storing data in a plurality of logical pages (L-Pages), each of the plurality of L-Pages being associated with an L-Page number that is configured to enable the controller to logically reference data stored in one or more of the physical pages;

maintaining a logical-to-physical address translation map in the volatile memory, the translation map being configured to enable determination of a physical location, within one or more of the physical pages, of the data stored in each L-Page;

maintaining a plurality of journals defining physical-to-logical correspondences in the plurality of non-volatile memory devices, each of the plurality of journals being

associated with a journal number, each journal covering a pre-determined range of physical pages and each comprising a plurality of journal entries, each entry being configured to associate one or more physical pages to each L-Page; and

reading the plurality of journals upon startup and rebuilding the translation map stored in volatile memory based upon the read entries in the plurality of journals.

24. The method of claim 23, further comprising creating a new entry in one of the plurality of journals upon an update to one of the plurality of L-Pages.

25. The method of claim 24, further comprising triggering write operations to the non-volatile memory devices to maintain a power-safe copy of translation data based on newly created journal entries instead of saving at least portions of the translation map, such that a write amplification is reduced.

26. The method of claim 24, wherein the new entry indicates a physical location, within a physical page, of a start of the updated L-Page.

27. The method of claim 24, further comprising updating a free space accounting by an amount corresponding to a length of the L-Page prior to being updated.

28. The method of claim 23, wherein storing comprises storing at least one of the plurality of L-Pages at a location that is unaligned with physical page boundaries.

29. The method of claim 23, wherein the physical pages are implemented as Error Correcting Code (ECC) pages (E-Pages) and wherein the plurality of devices comprises a plurality of flash memory blocks, each flash memory block comprising a plurality of flash memory pages (F-Pages), each of the F-Pages comprising a plurality of the E-Pages, each of the plurality of E-Pages being stored at a predetermined physical location within the plurality of devices.

30. The method of claim 23, further comprising updating the translation map with the physical location, within one or more physical pages, of the data referenced by the L-Page number of the updated L-Page.

31. The method of claim 23, wherein the plurality of devices comprises a plurality of super blocks (S-Blocks), each comprising one or more flash memory blocks per device and wherein each of the plurality of journal entries is configured to associated one or more of the physical pages of the S-Block to each L-Page.

32. The method of claim 31, further comprising:

selecting an S-Block to garbage collect;

comparing each entry in a journal for the selected S-Block to entries in the translation map and designating entries that match as valid and entries that do not match as invalid;

reading the L-Pages corresponding to the valid entries;

writing the read L-Pages to respective physical addresses within the plurality of non-volatile memory devices;

updating the translation map for the valid entries to point to the respective physical addresses; and

generating new journal entries for the entries for which the translation map was updated.

33. The method of claim 31, further comprising

selecting one of the plurality of S-Blocks to garbage collect;

reading the physical pages of the selected S-Block;

comparing L-Page numbers in the read physical pages of the selected S-Block to entries in the translation map and designating entries that match as valid and entries that do not match as invalid;

writing the L-Pages corresponding to the valid entries to respective physical addresses within the plurality of non-volatile memory devices;

updating the translation map for the valid entries to point to the respective

physical addresses; and

generating new journal entries for the entries for which the translation map was updated.

34. The method of claim 32, wherein selecting comprises weighing free space and program erase (PE) count in determining which S-Block to select.

35. The method of claim 34, wherein the journal number comprises a predetermined number of most significant bits of an address of a first physical page covered by the journal.

36. The method of claim 23, wherein each of the plurality of journal entries comprises:

an L-Page number, and

a physical address location.

37. The method of claim 23, wherein each of the plurality of journal entries comprises:

an L-Page number;

a physical address location of a physical page, and

an L-Page size.

38. The method of claim 23, wherein each of the plurality of journal entries comprises:

a predetermined number of least significant bits of an address of a physical page that includes a start of an L-Page;

an address;

an L-Page size; and

an offset into the physical page.

39. The method of claim 23, further comprising selectively compressing the

plurality of L-Pages such that the plurality of L-Pages vary in size and wherein the plurality of journals are configured to reference a greater number of L-Pages of smaller size or a lesser number of L-Pages of greater size.

40. The method of claim 23, wherein reading the plurality of journals upon startup and rebuilding the translation map comprises reading the plurality of journals upon startup in a predetermined sequential order and rebuilding the translation map stored in volatile memory based upon the read plurality of journals.

41. The method of claim 23, wherein the controller is further configured to build a journal map based on the plurality of journals.

42. The method of claim 23, further comprising:

reading the plurality of journals upon startup in a predetermined sequential order;

building a map of the journals stored in the non-volatile memory devices based upon the sequentially read plurality of journals, and

storing the built map of the journals in the volatile memory.

43. The method of claim 23 wherein, among journal entries associated with a given L-Page, only a last-in-time updated journal entry associated with the given L-Page is valid.

44. The method of claim 23, further comprising maintaining a system journal map of the plurality of journals in the volatile memory, each entry in the system journal map pointing to a location in the non-volatile memory devices where one of the plurality of journals is stored.

45. A data storage device controller, comprising:

a processor configured to couple to a volatile memory and to a plurality of memory devices, each of the plurality of memory devices being configured to store a plurality of physical pages at a predetermined physical location within the plurality of

non-volatile devices, the processor being further configured to program data to and read data from the plurality of memory devices, the data being stored in a plurality of logical pages (L-Pages), each of the plurality of L-Pages being associated with an L-Page number that is configured to enable the processor to logically reference data stored in one or more of the physical pages, the volatile memory being configured to store a logical-to-physical address translation map configured to enable the processor to determine a physical location, within one or more physical pages, of the data stored in each L-Page,

wherein the processor is configured to maintain, in the plurality of non-volatile memory devices, a plurality of journals defining physical-to-logical correspondences, each of the plurality of journals being associated with a journal number, each journal covering a pre-determined range of physical pages and comprising a plurality of journal entries, each entry being configured to associate one or more physical pages to each L-Page, wherein the processor is further configured to read the plurality of journals upon startup and rebuild the address translation map stored in the volatile memory from the read plurality of journals.

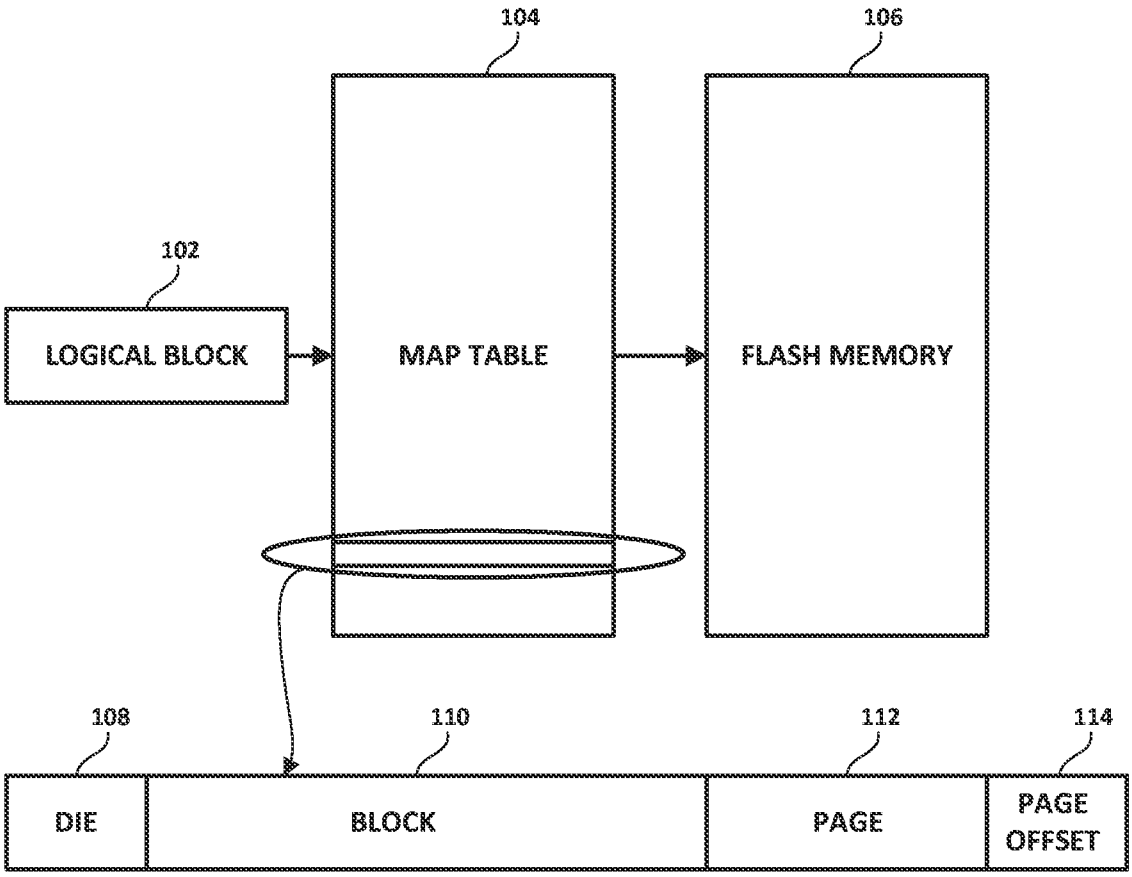


FIG. 1

(Prior Art)

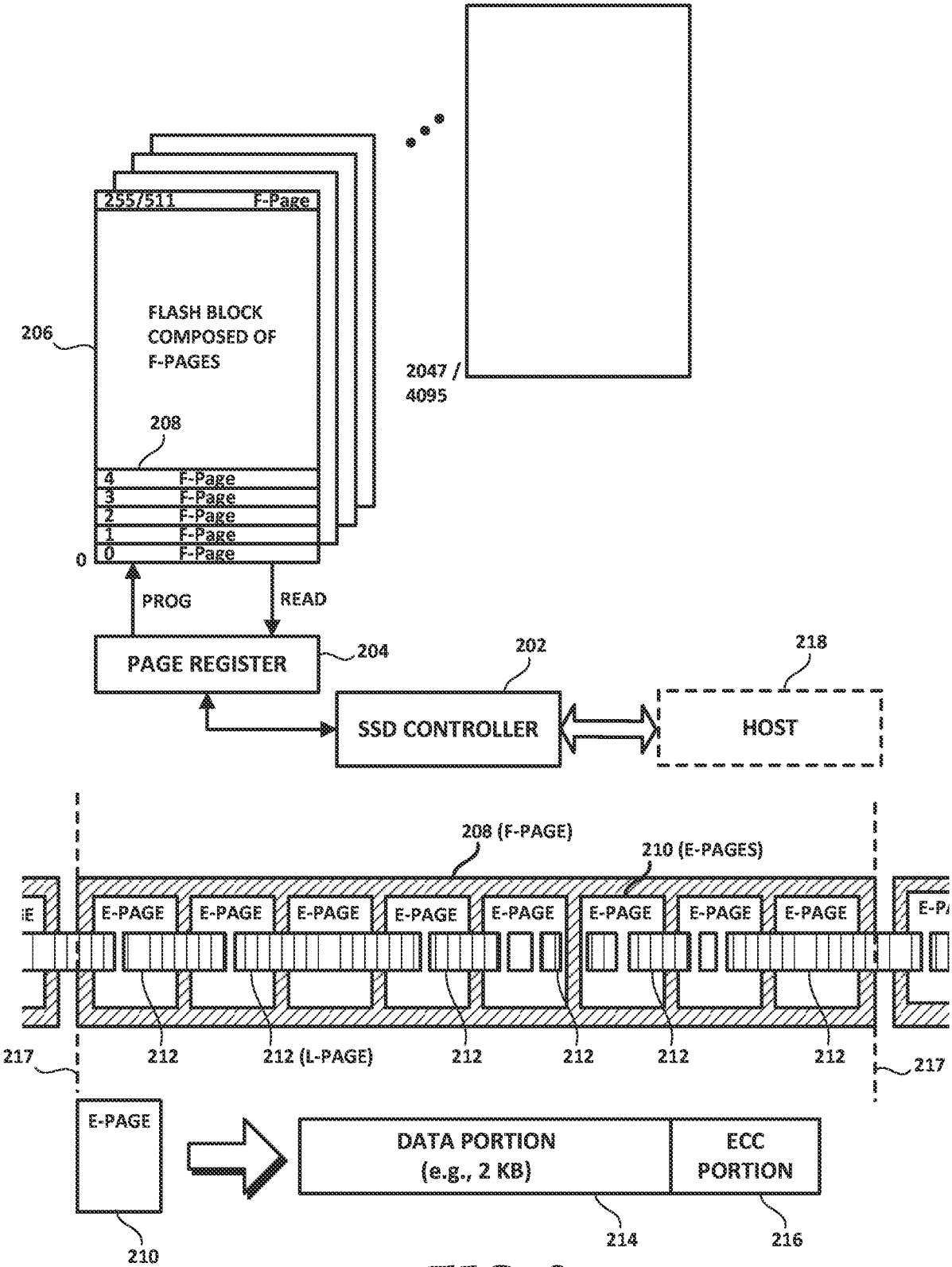
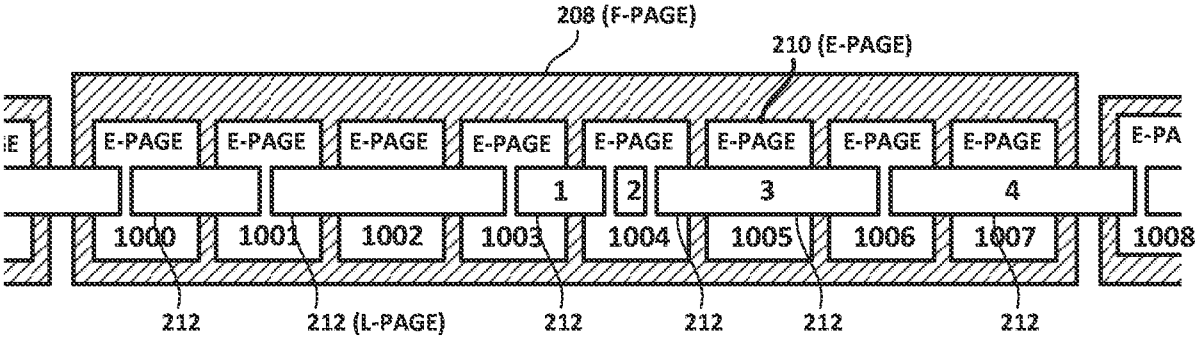

255/511 F-Page
FLASH BLOCK COMPOSED OF F-PAGES
206
208
4 F-Page
3 F-Page
2 F-Page
1 F-Page
0 F-Page
0
2047 / 4095
PROG
READ
PAGE REGISTER
204
202
SSD CONTROLLER
218
HOST
208 (F-PAGE)
210 (E-PAGES)
E-PAGE
212
212 (L-PAGE)
217
E-PAGE
DATA PORTION (e.g., 2 KB)
ECC PORTION
210
214
216

FIG. 2



L-PAGE #				E-PAGES NEEDED
L-Page 1 =	E-Page 1003	Offset 800	Len 1,624	1003,1004
L-Page 2 =	E-Page 1004	Offset 400	Len 696	1004
L-Page 3 =	E-Page 1004	Offset 1,120	Len 4,096	1004, 1005, 1006
L-Page 4 =	E-Page 1006	Offset 1,144	Len 3,128	1006, 1007, 1008

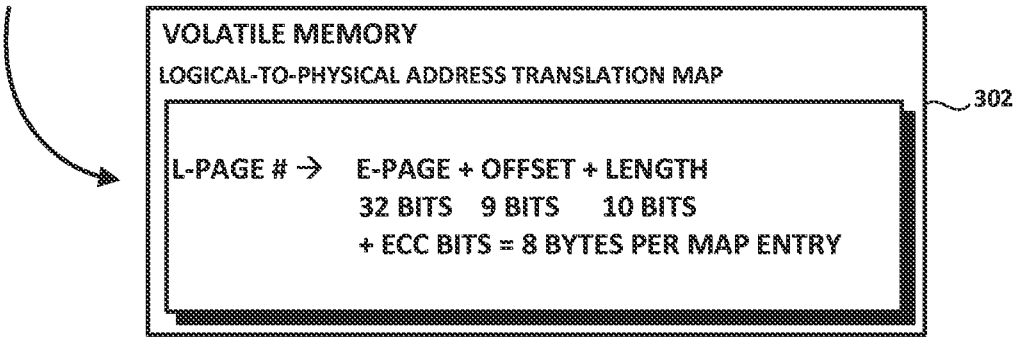


FIG. 3

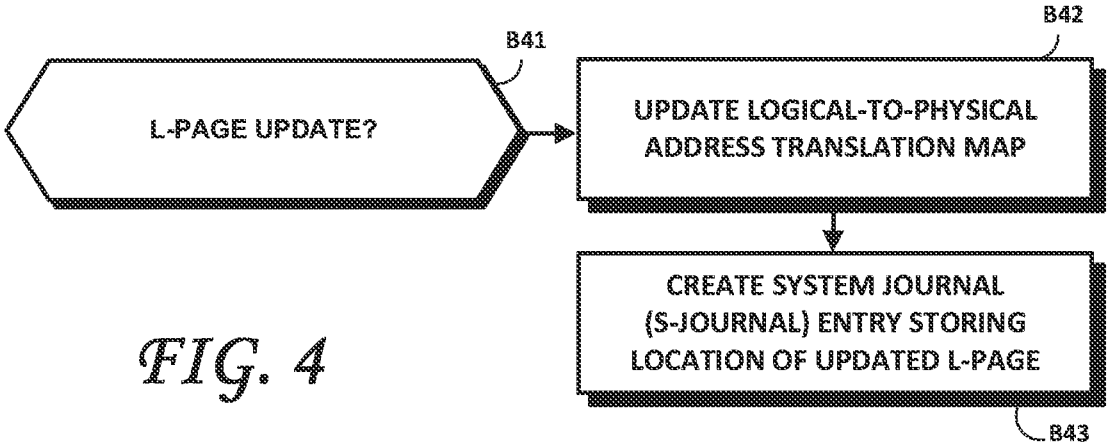


FIG. 4

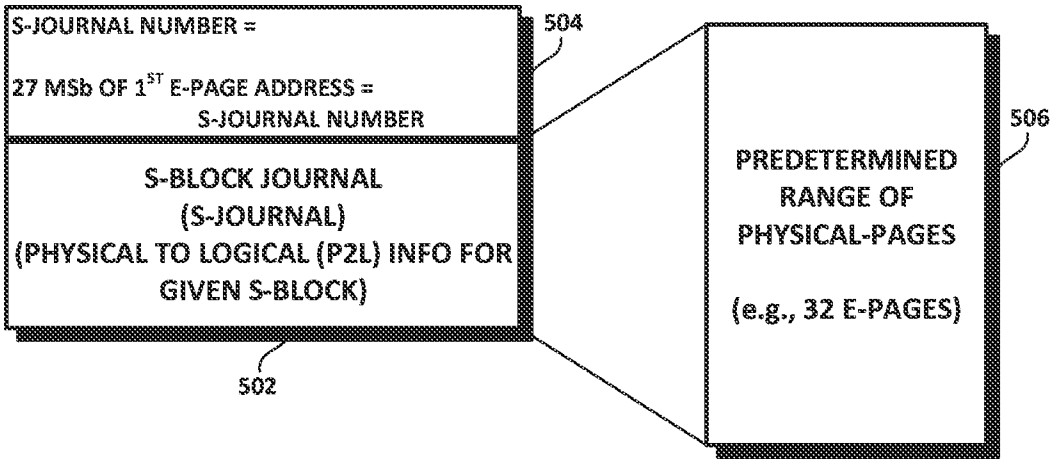

S-JOURNAL NUMBER =
27 MSb OF 1ST E-PAGE ADDRESS =
S-JOURNAL NUMBER
S-BLOCK JOURNAL
(S-JOURNAL)
(PHYSICAL TO LOGICAL (P2L) INFO FOR
GIVEN S-BLOCK)
502
504
PREDETERMINED
RANGE OF
PHYSICAL-PAGES
(e.g., 32 E-PAGES)
506

FIG. 5

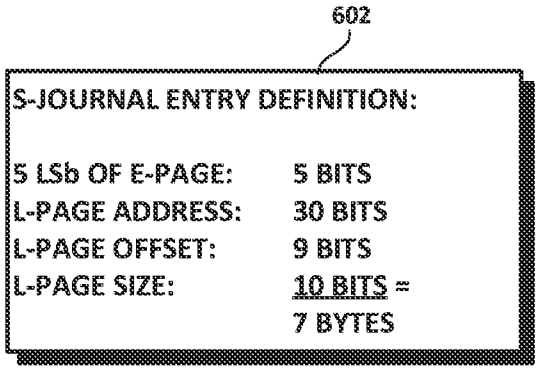

602
S-JOURNAL ENTRY DEFINITION:
5 LSb OF E-PAGE: 5 BITS
L-PAGE ADDRESS: 30 BITS
L-PAGE OFFSET: 9 BITS
L-PAGE SIZE: 10 BITS =
7 BYTES

FIG. 6

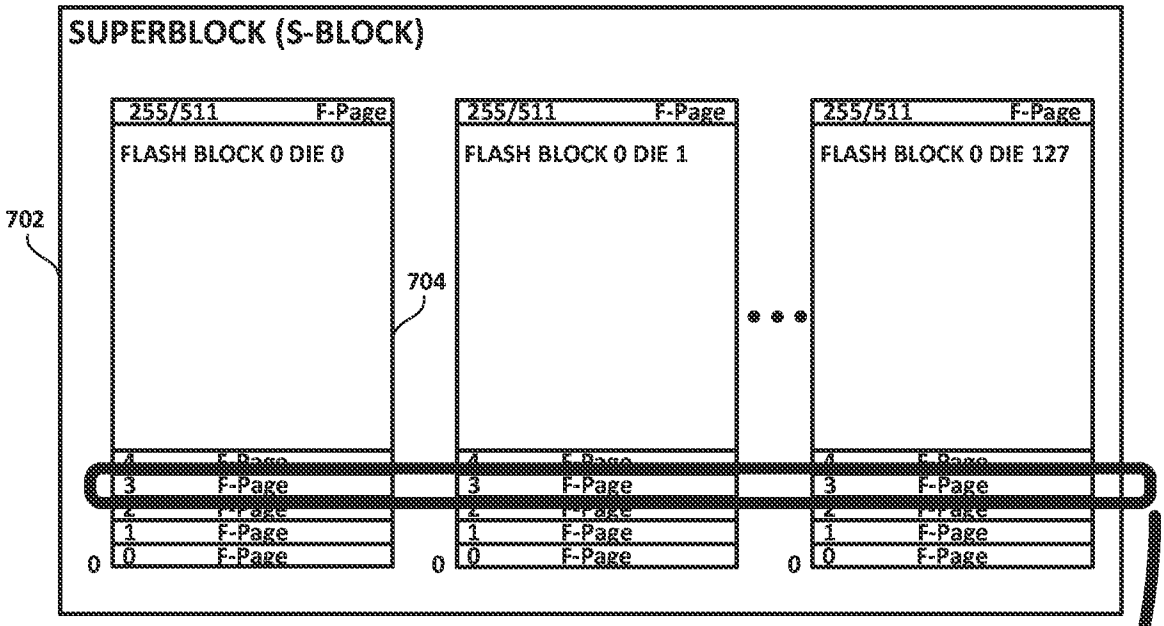

SUPERBLOCK (S-BLOCK)
255/511 F-Page
FLASH BLOCK 0 DIE 0
FLASH BLOCK 0 DIE 1
FLASH BLOCK 0 DIE 127
702
704
3 F-Page
1 F-Page
0 F-Page

FIG. 7

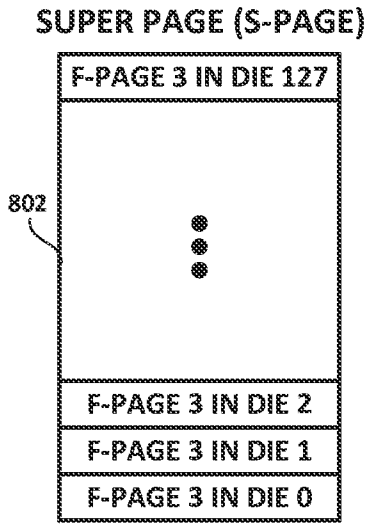

SUPER PAGE (S-PAGE)
F-PAGE 3 IN DIE 127
802
F-PAGE 3 IN DIE 2
F-PAGE 3 IN DIE 1
F-PAGE 3 IN DIE 0

FIG. 8

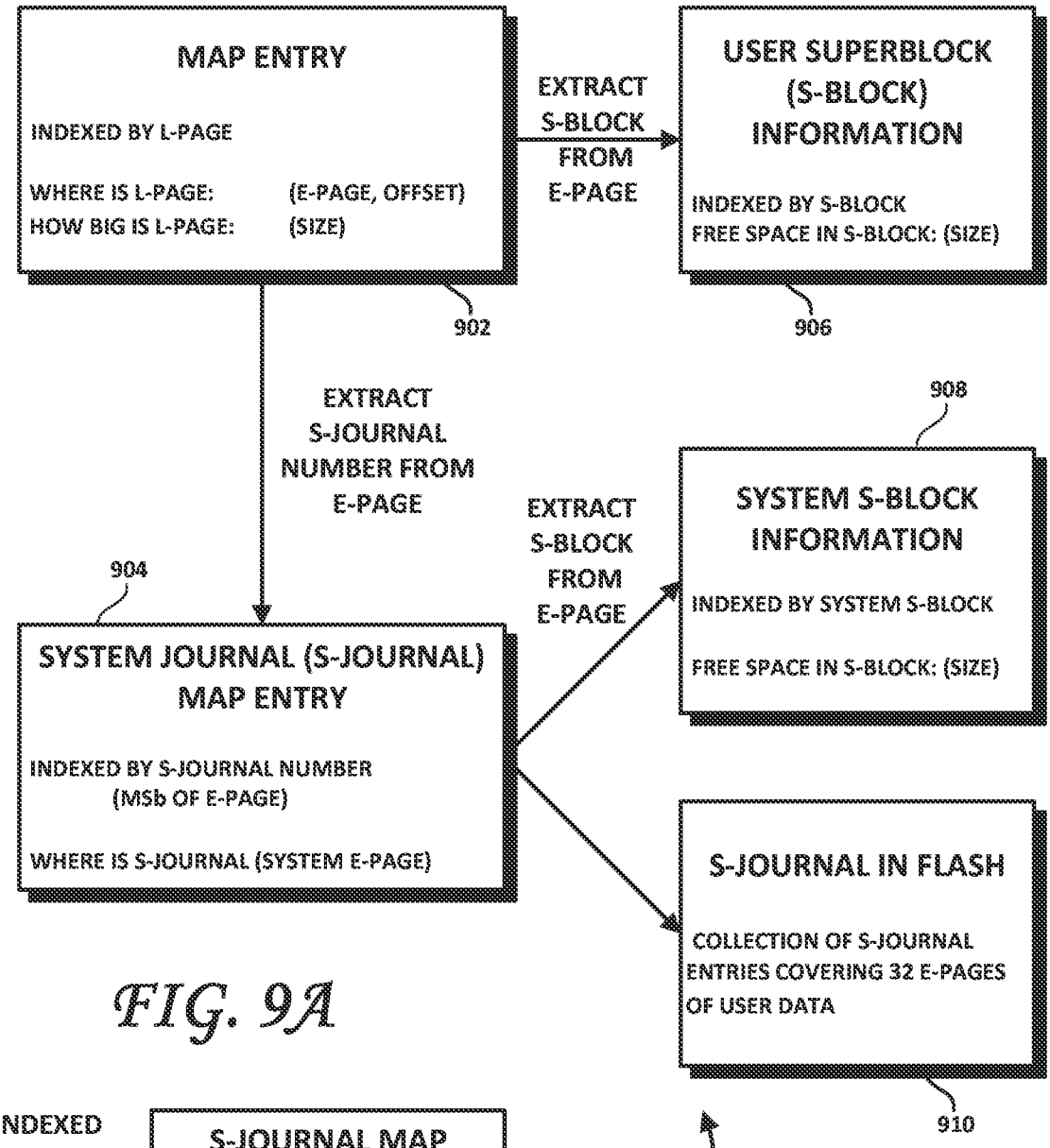

MAP ENTRY
INDEXED BY L-PAGE
WHERE IS L-PAGE: (E-PAGE, OFFSET)
HOW BIG IS L-PAGE: (SIZE)
902
EXTRACT S-BLOCK FROM E-PAGE
USER SUPERBLOCK (S-BLOCK) INFORMATION
INDEXED BY S-BLOCK
FREE SPACE IN S-BLOCK: (SIZE)
906
EXTRACT S-JOURNAL NUMBER FROM E-PAGE
908
SYSTEM S-BLOCK INFORMATION
INDEXED BY SYSTEM S-BLOCK
FREE SPACE IN S-BLOCK: (SIZE)
EXTRACT S-BLOCK FROM E-PAGE
904
SYSTEM JOURNAL (S-JOURNAL) MAP ENTRY
INDEXED BY S-JOURNAL NUMBER (MSb OF E-PAGE)
WHERE IS S-JOURNAL (SYSTEM E-PAGE)
S-JOURNAL IN FLASH
COLLECTION OF S-JOURNAL ENTRIES COVERING 32 E-PAGES OF USER DATA
910

FIG. 9A

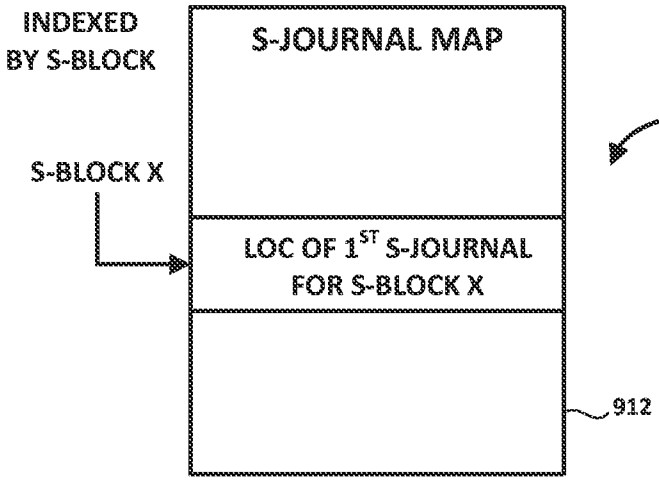

INDEXED BY S-BLOCK
S-JOURNAL MAP
S-BLOCK X
LOC OF 1ST S-JOURNAL FOR S-BLOCK X
912

FIG. 9B

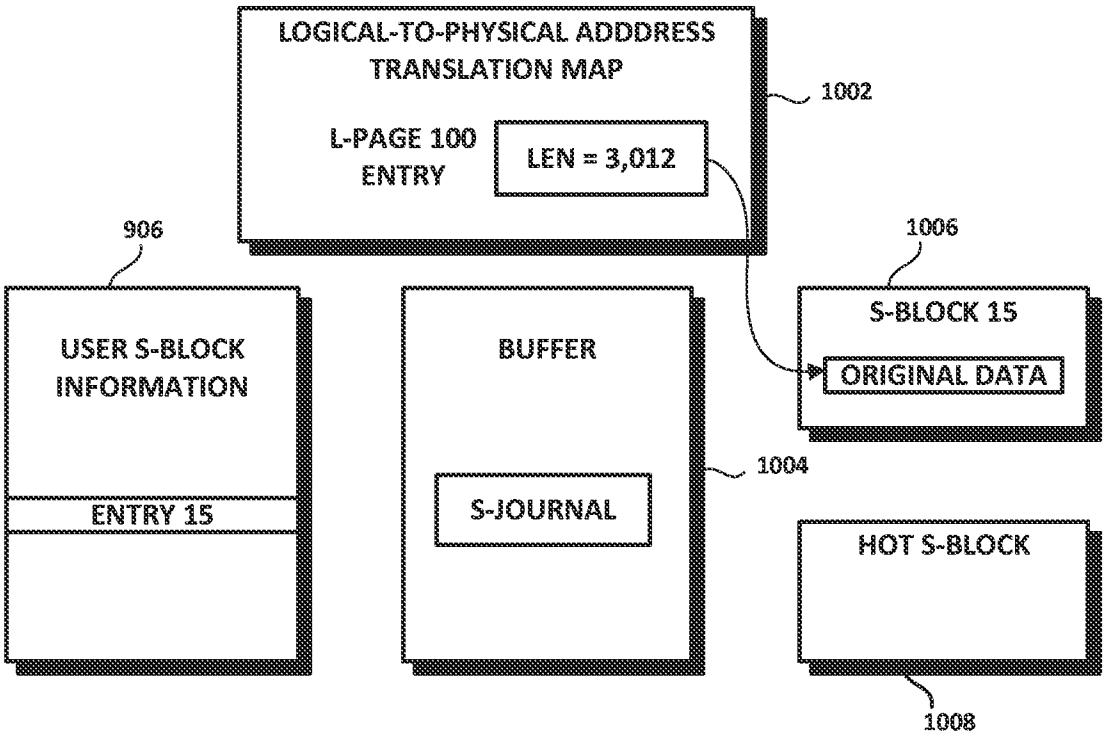

LOGICAL-TO-PHYSICAL ADDDRESS TRANSLATION MAP
1002
L-PAGE 100 ENTRY
LEN = 3,012
906
1006
USER S-BLOCK INFORMATION
ENTRY 15
BUFFER
S-JOURNAL
1004
S-BLOCK 15
ORIGINAL DATA
HOT S-BLOCK
1008

FIG. 10

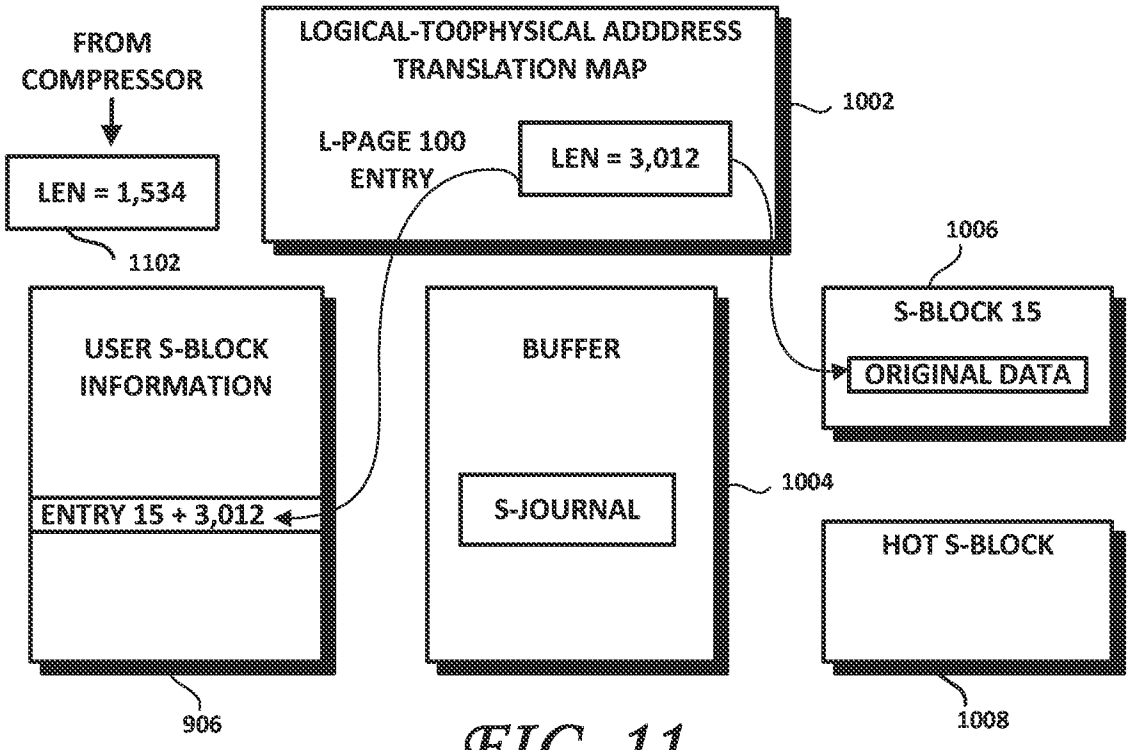

FROM COMPRESSOR
LEN = 1,534
1102
LOGICAL-TO0PHYSICAL ADDDRESS TRANSLATION MAP
1002
L-PAGE 100 ENTRY
LEN = 3,012
1006
USER S-BLOCK INFORMATION
ENTRY 15 + 3,012
BUFFER
S-JOURNAL
1004
S-BLOCK 15
ORIGINAL DATA
HOT S-BLOCK
906
1008

FIG. 11

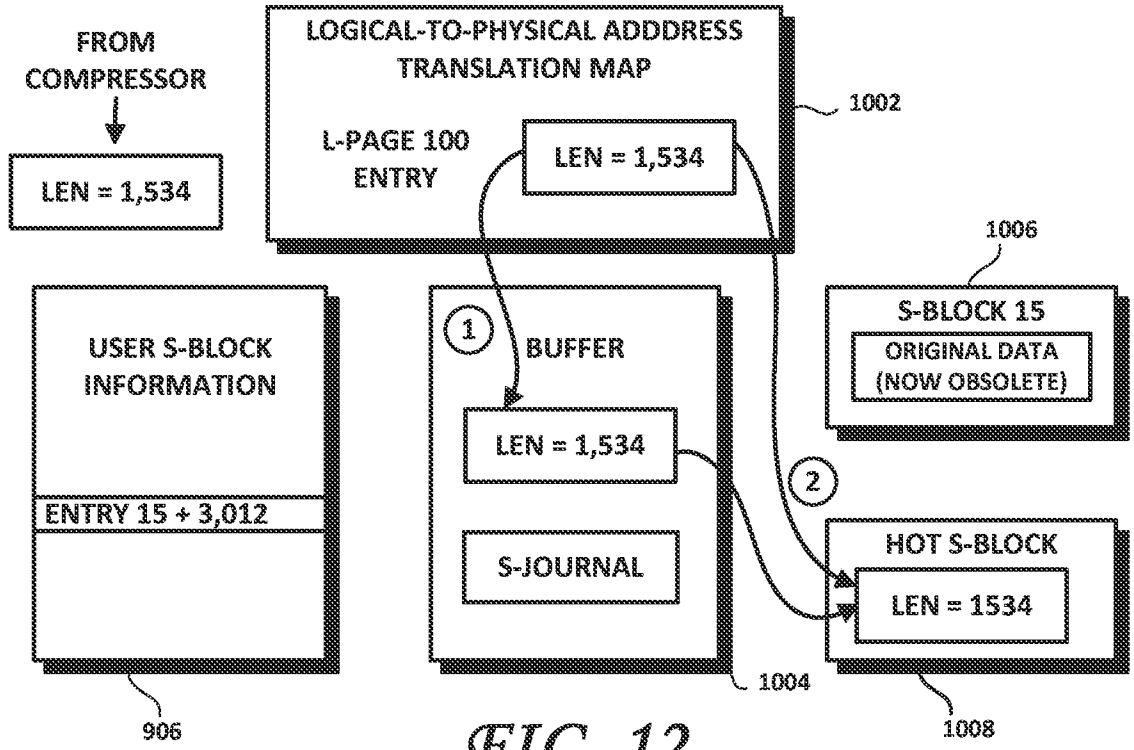

FROM COMPRESSOR
LEN = 1,534
LOGICAL-TO-PHYSICAL ADDDRESS TRANSLATION MAP
L-PAGE 100 ENTRY
LEN = 1,534
1002
1006
S-BLOCK 15
ORIGINAL DATA (NOW OBSOLETE)
USER S-BLOCK INFORMATION
ENTRY 15 + 3,012
1
BUFFER
LEN = 1,534
S-JOURNAL
2
HOT S-BLOCK
LEN = 1534
1004
906
1008

FIG. 12

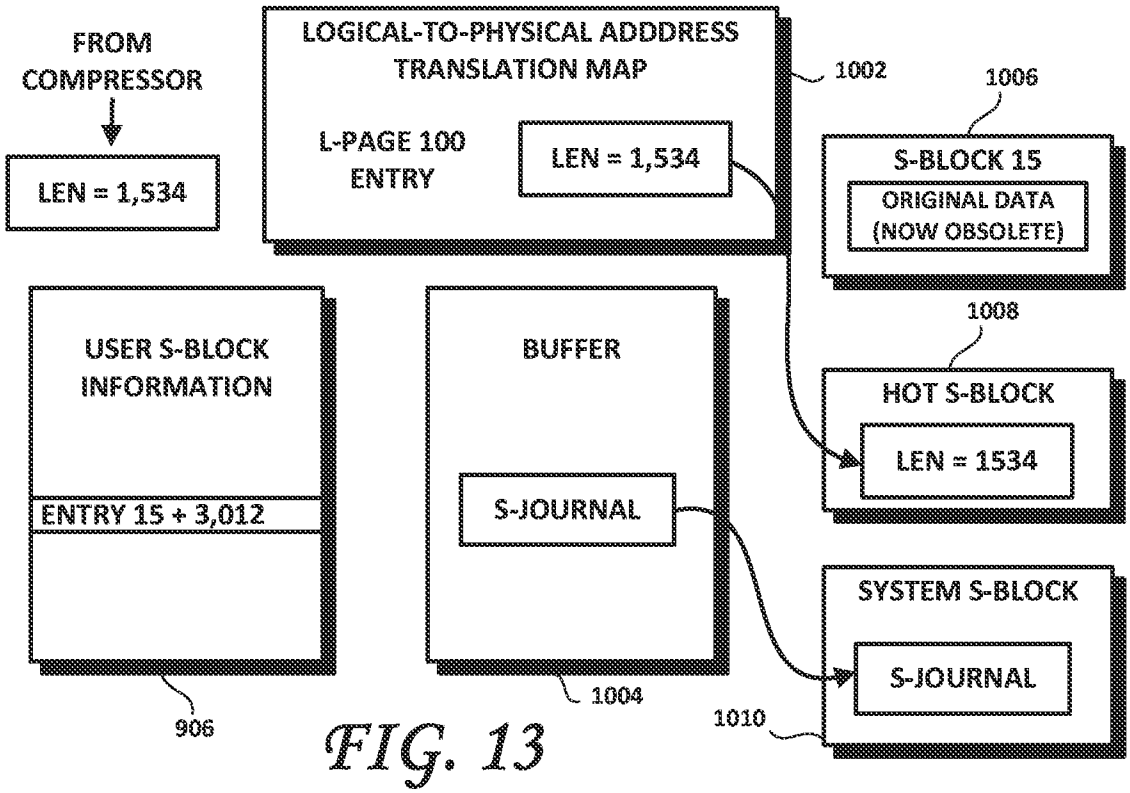

FROM COMPRESSOR
LEN = 1,534
LOGICAL-TO-PHYSICAL ADDDRESS TRANSLATION MAP
L-PAGE 100 ENTRY
LEN = 1,534
1002
1006
S-BLOCK 15
ORIGINAL DATA (NOW OBSOLETE)
1008
USER S-BLOCK INFORMATION
ENTRY 15 + 3,012
BUFFER
S-JOURNAL
HOT S-BLOCK
LEN = 1534
SYSTEM S-BLOCK
S-JOURNAL
1004
906
1010

FIG. 13

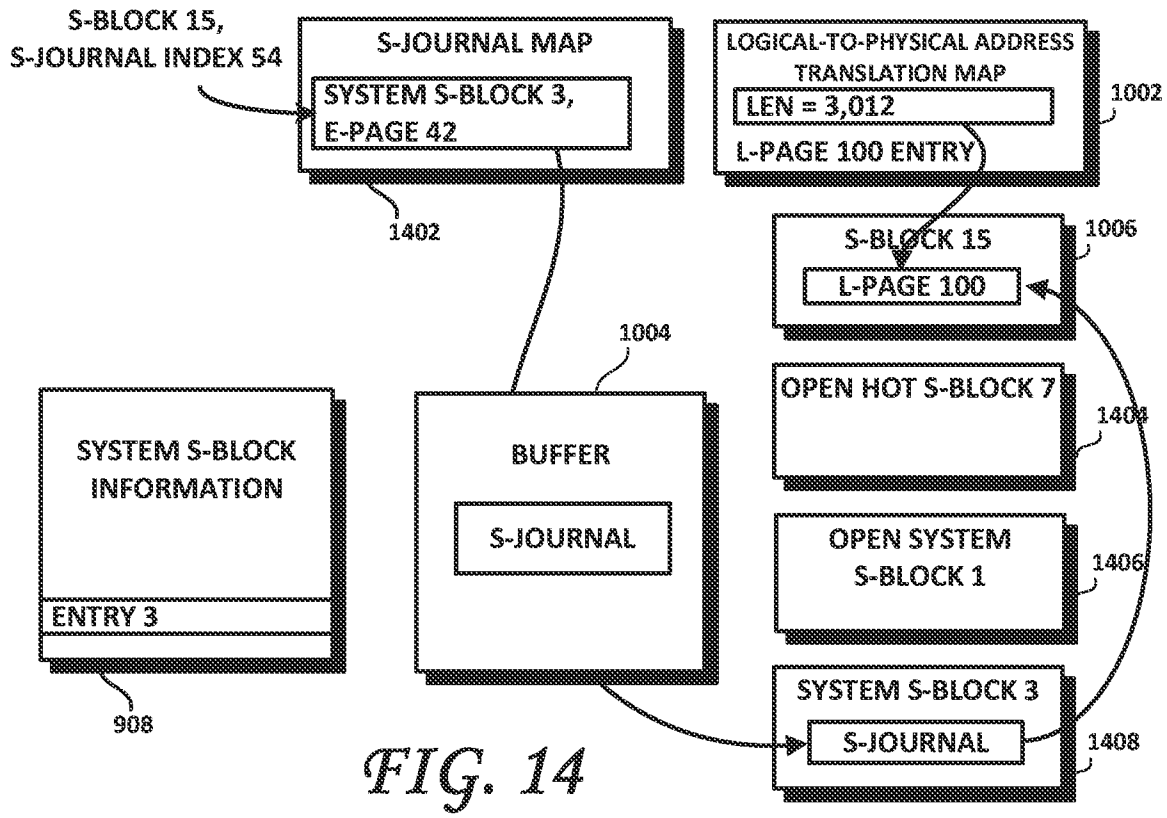

S-BLOCK 15,
S-JOURNAL INDEX 54
S-JOURNAL MAP
SYSTEM S-BLOCK 3,
E-PAGE 42
1402
LOGICAL-TO-PHYSICAL ADDRESS
TRANSLATION MAP
LEN = 3,012
L-PAGE 100 ENTRY
1002
S-BLOCK 15
L-PAGE 100
1006
1004
OPEN HOT S-BLOCK 7
1404
SYSTEM S-BLOCK
INFORMATION
BUFFER
S-JOURNAL
OPEN SYSTEM
S-BLOCK 1
1406
ENTRY 3
SYSTEM S-BLOCK 3
S-JOURNAL
1408
908

FIG. 14

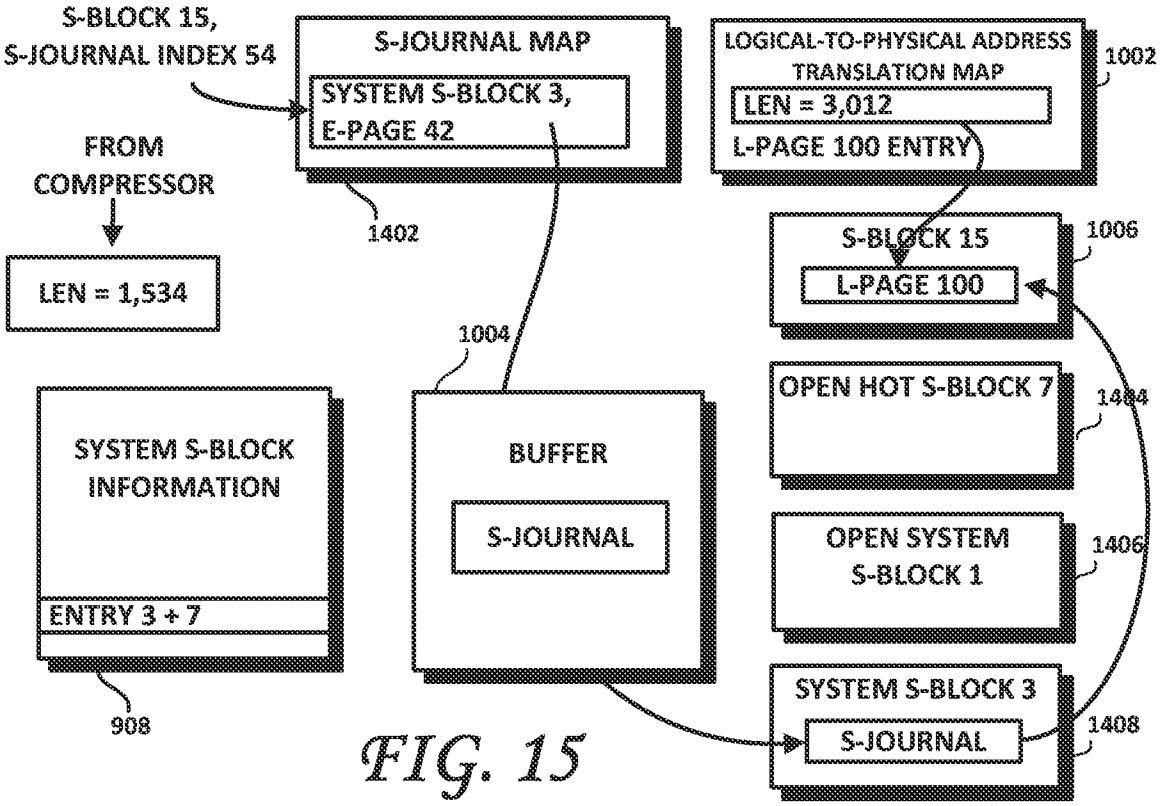

S-BLOCK 15,
S-JOURNAL INDEX 54
S-JOURNAL MAP
SYSTEM S-BLOCK 3,
E-PAGE 42
1402
LOGICAL-TO-PHYSICAL ADDRESS
TRANSLATION MAP
LEN = 3,012
L-PAGE 100 ENTRY
1002
FROM
COMPRESSOR
LEN = 1,534
S-BLOCK 15
L-PAGE 100
1006
1004
OPEN HOT S-BLOCK 7
1404
SYSTEM S-BLOCK
INFORMATION
BUFFER
S-JOURNAL
OPEN SYSTEM
S-BLOCK 1
1406
ENTRY 3 + 7
SYSTEM S-BLOCK 3
S-JOURNAL
1408
908

FIG. 15

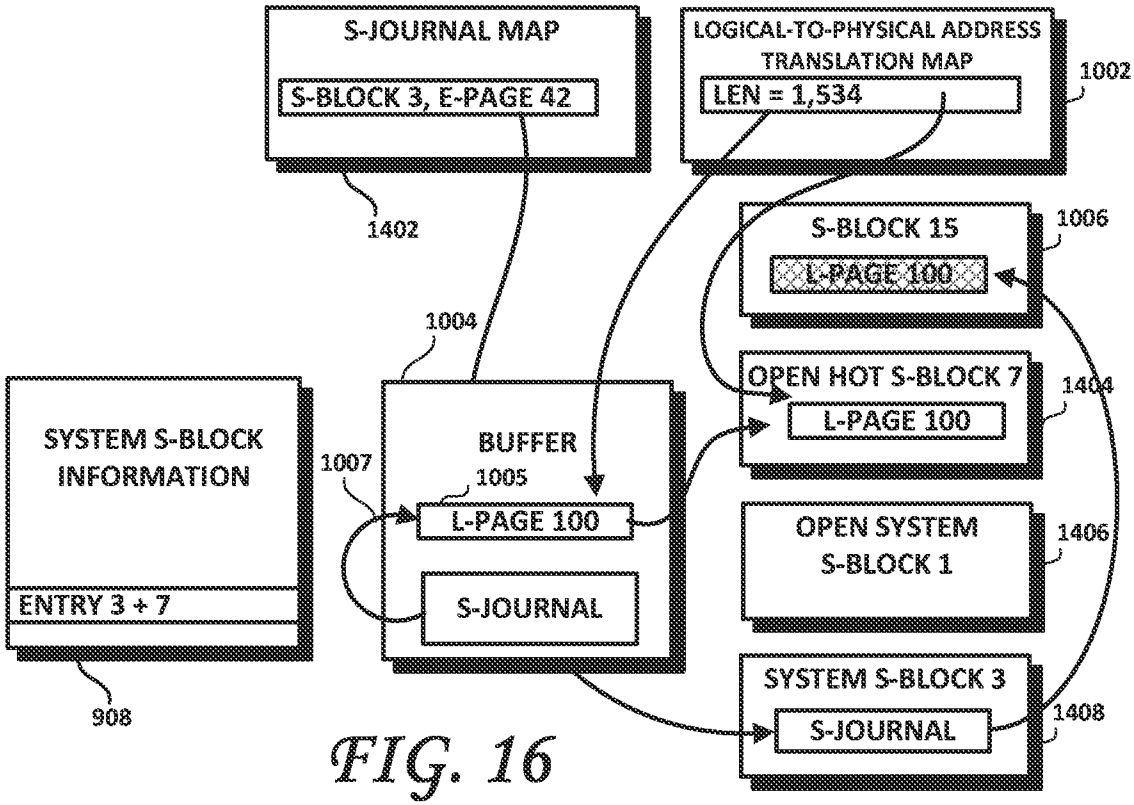

S-JOURNAL MAP
S-BLOCK 3, E-PAGE 42
1402
LOGICAL-TO-PHYSICAL ADDRESS TRANSLATION MAP
LEN = 1,534
1002
S-BLOCK 15
L-PAGE 100
1006
1004
OPEN HOT S-BLOCK 7
L-PAGE 100
1404
SYSTEM S-BLOCK INFORMATION
1007
BUFFER
1005
L-PAGE 100
S-JOURNAL
OPEN SYSTEM S-BLOCK 1
1406
ENTRY 3 ÷ 7
908
SYSTEM S-BLOCK 3
S-JOURNAL
1408

FIG. 16

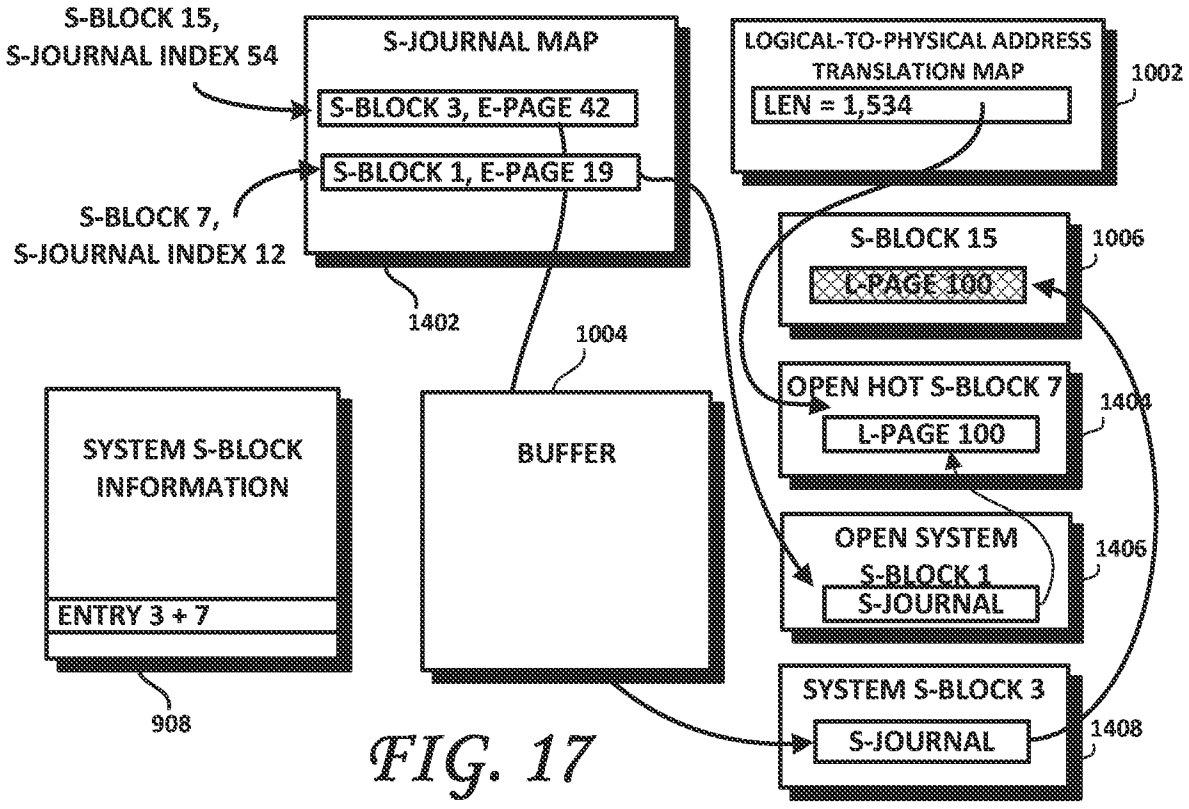

S-BLOCK 15, S-JOURNAL INDEX 54
S-JOURNAL MAP
S-BLOCK 3, E-PAGE 42
S-BLOCK 1, E-PAGE 19
S-BLOCK 7, S-JOURNAL INDEX 12
1402
LOGICAL-TO-PHYSICAL ADDRESS TRANSLATION MAP
LEN = 1,534
1002
S-BLOCK 15
L-PAGE 100
1006
1004
OPEN HOT S-BLOCK 7
L-PAGE 100
1404
SYSTEM S-BLOCK INFORMATION
BUFFER
OPEN SYSTEM S-BLOCK 1
S-JOURNAL
1406
ENTRY 3 ÷ 7
908
SYSTEM S-BLOCK 3
S-JOURNAL
1408

FIG. 17

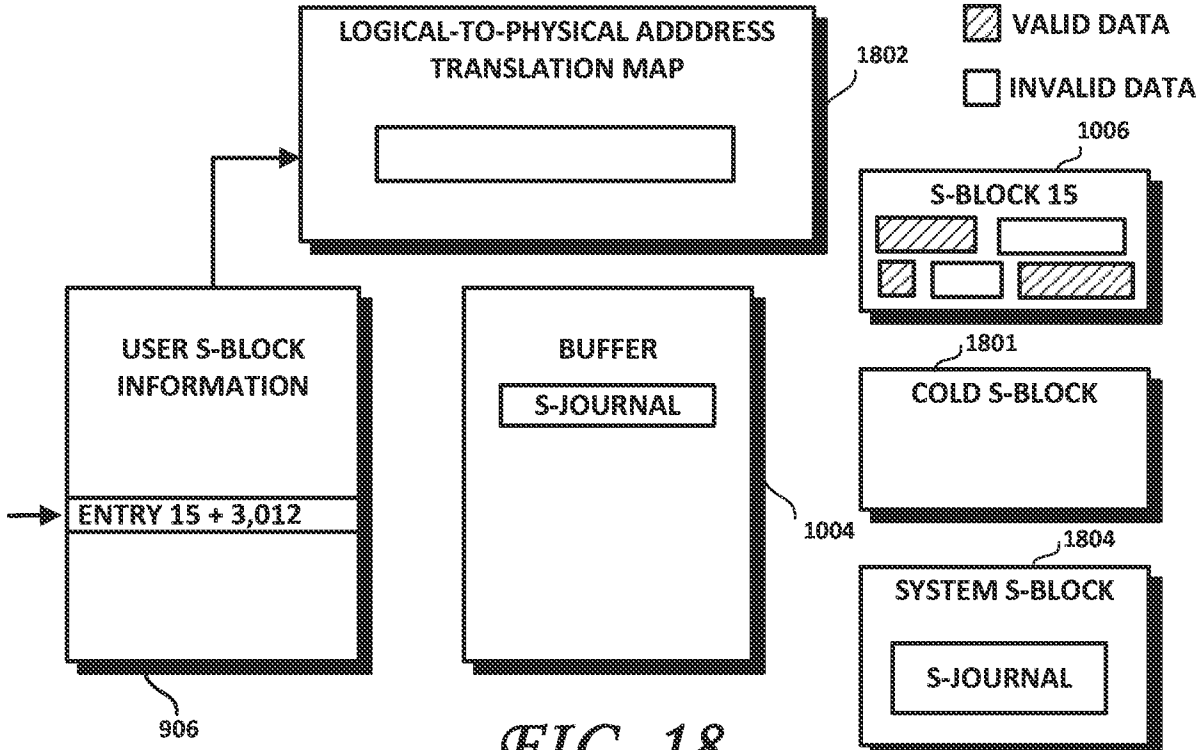

LOGICAL-TO-PHYSICAL ADDDRESS TRANSLATION MAP
1802
VALID DATA
INVALID DATA
1006
S-BLOCK 15
USER S-BLOCK INFORMATION
ENTRY 15 + 3,012
BUFFER
S-JOURNAL
1801
COLD S-BLOCK
1004
1804
SYSTEM S-BLOCK
S-JOURNAL
906

FIG. 18

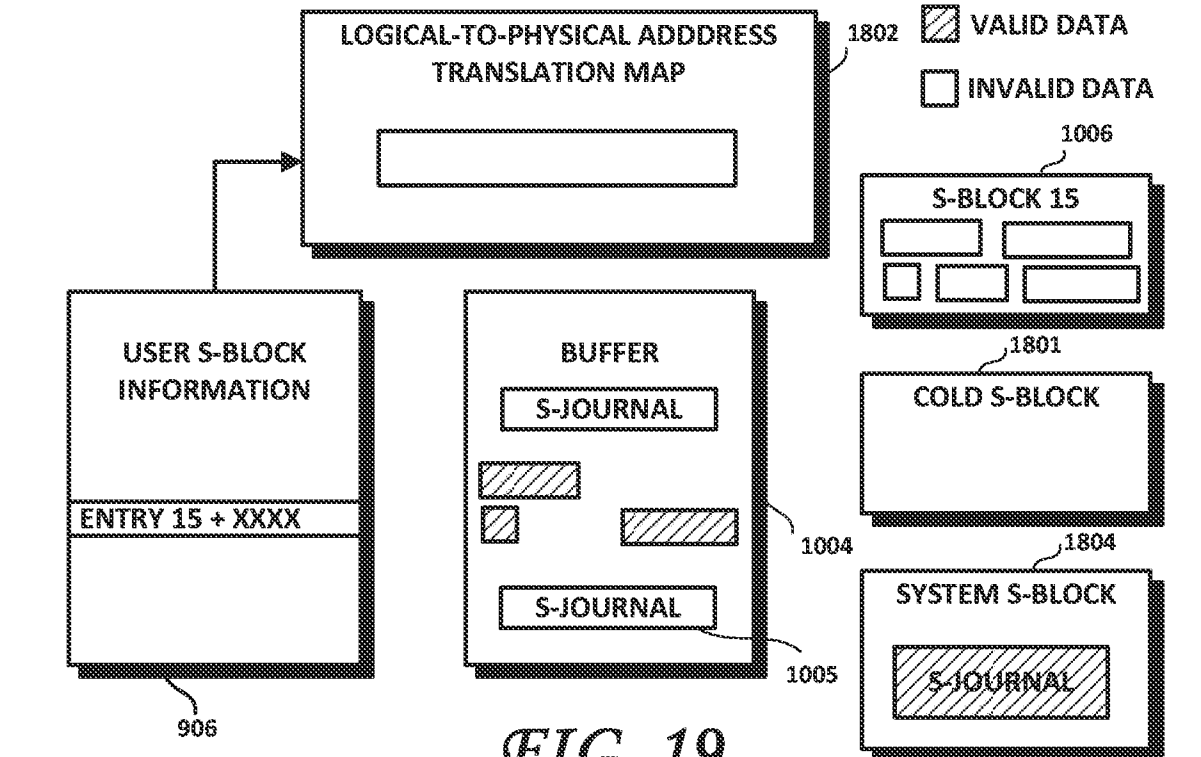

LOGICAL-TO-PHYSICAL ADDDRESS TRANSLATION MAP
1802
VALID DATA
INVALID DATA
1006
S-BLOCK 15
USER S-BLOCK INFORMATION
ENTRY 15 + XXXX
BUFFER
S-JOURNAL
1801
COLD S-BLOCK
1004
1804
SYSTEM S-BLOCK
S-JOURNAL
1005
S-JOURNAL
906

FIG. 19

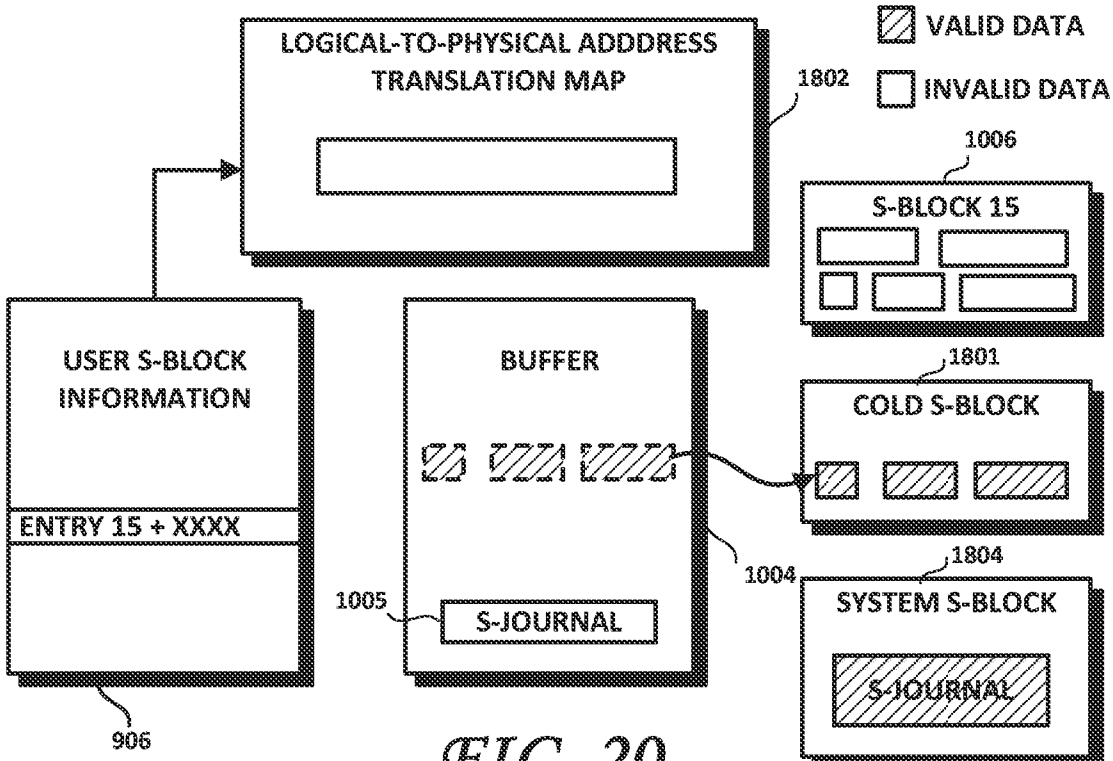

VALID DATA
INVALID DATA
LOGICAL-TO-PHYSICAL ADDDRESS TRANSLATION MAP
1802
1006
S-BLOCK 15
USER S-BLOCK INFORMATION
ENTRY 15 + XXXX
906
BUFFER
1005
S-JOURNAL
1801
COLD S-BLOCK
1004
1804
SYSTEM S-BLOCK
S-JOURNAL

FIG. 20

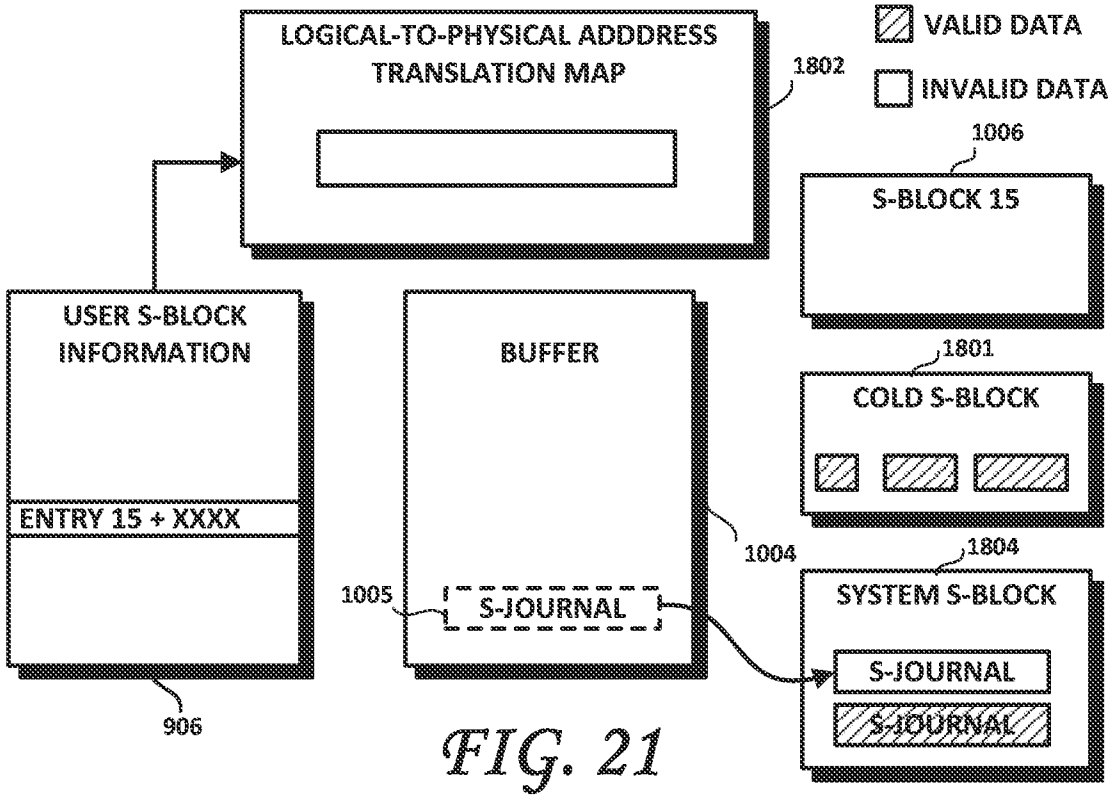

VALID DATA
INVALID DATA
LOGICAL-TO-PHYSICAL ADDDRESS TRANSLATION MAP
1802
1006
S-BLOCK 15
USER S-BLOCK INFORMATION
ENTRY 15 + XXXX
906
BUFFER
1005
S-JOURNAL
1801
COLD S-BLOCK
1004
1804
SYSTEM S-BLOCK
S-JOURNAL
S-JOURNAL

FIG. 21

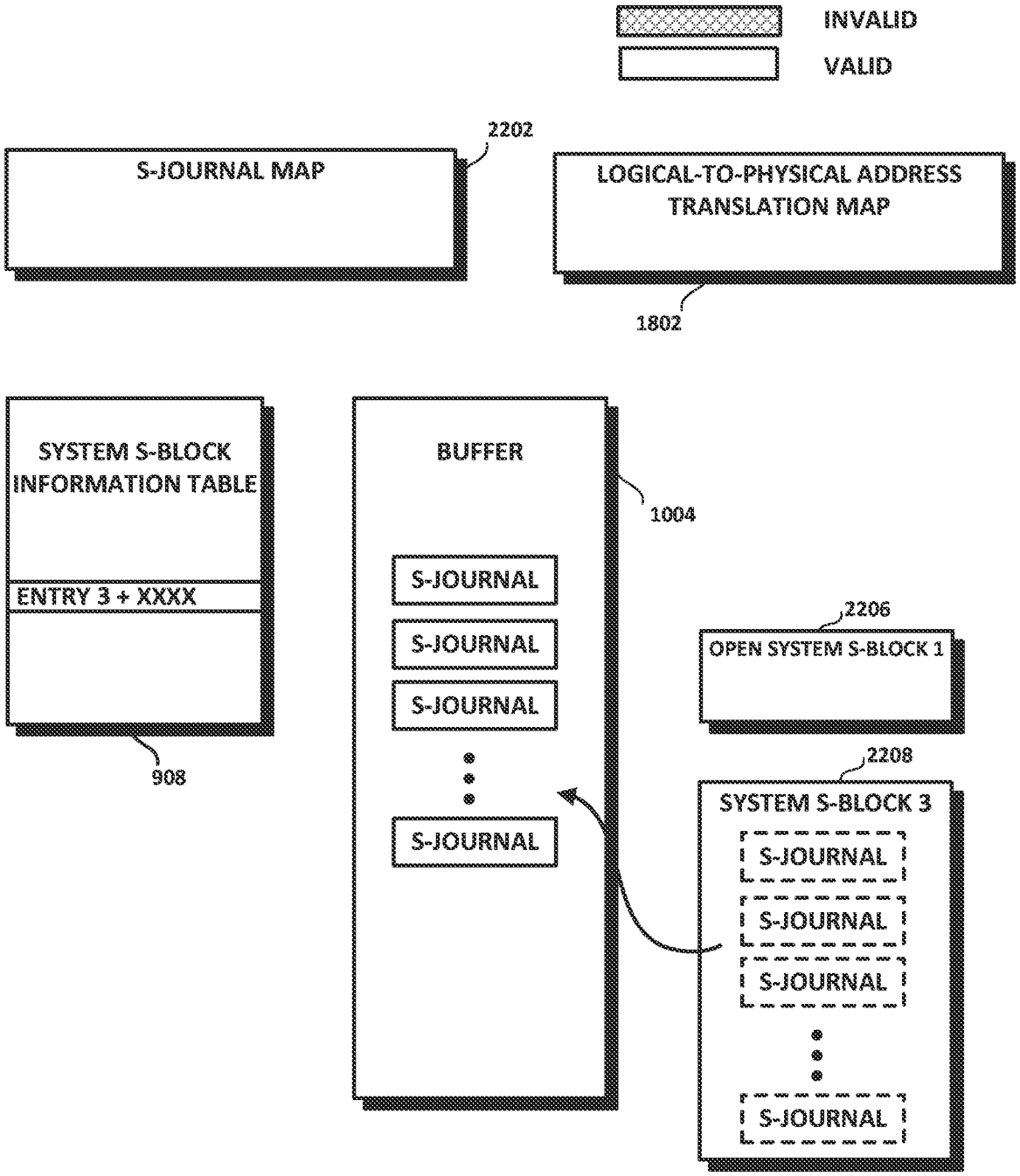

INVALID
VALID
2202
S-JOURNAL MAP
LOGICAL-TO-PHYSICAL ADDRESS TRANSLATION MAP
1802
SYSTEM S-BLOCK INFORMATION TABLE
ENTRY 3 + XXXX
908
BUFFER
1004
S-JOURNAL
S-JOURNAL
S-JOURNAL
S-JOURNAL
2206
OPEN SYSTEM S-BLOCK 1
2208
SYSTEM S-BLOCK 3
S-JOURNAL
S-JOURNAL
S-JOURNAL
S-JOURNAL

FIG. 22

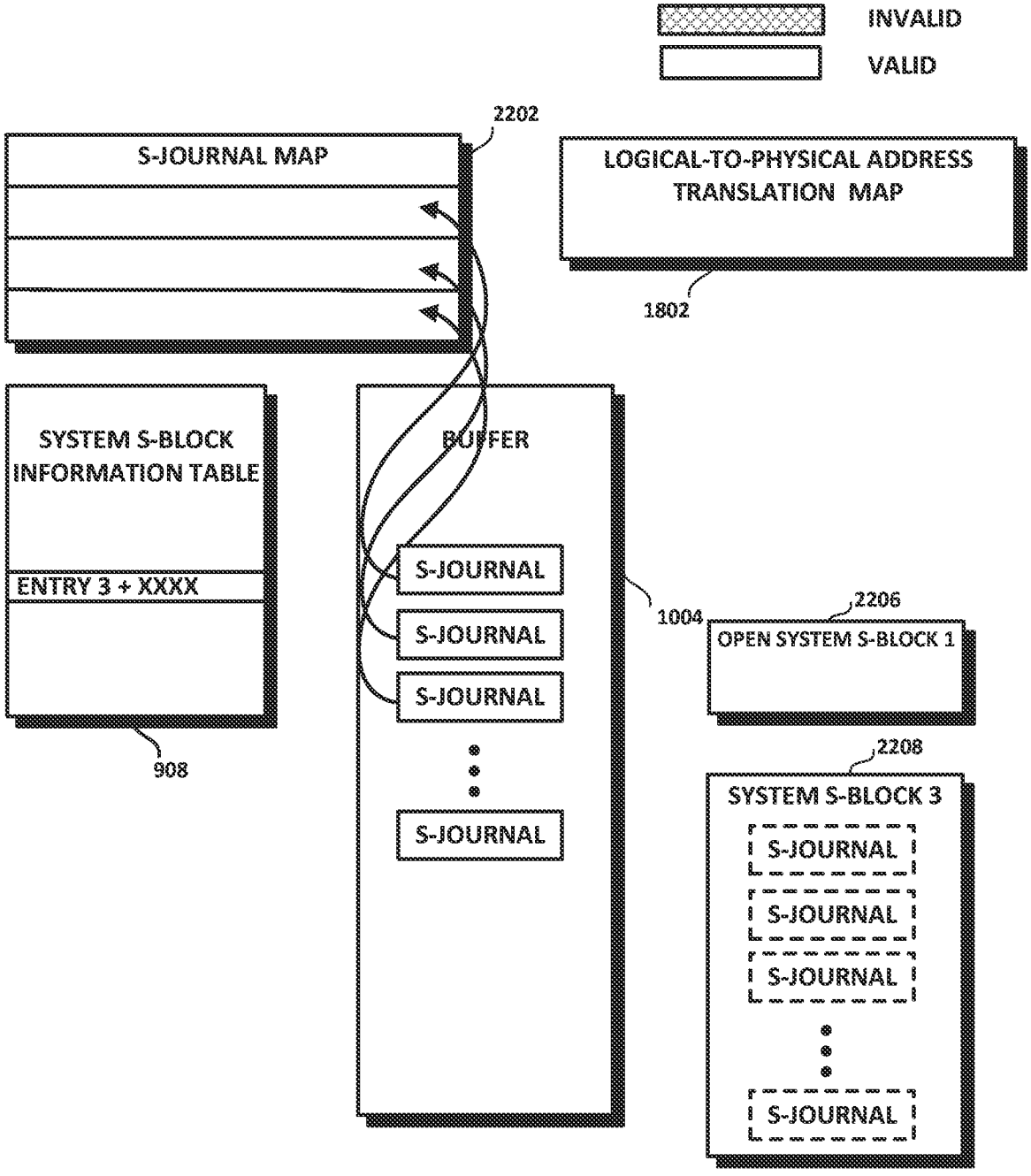

INVALID
VALID
2202
S-JOURNAL MAP
LOGICAL-TO-PHYSICAL ADDRESS TRANSLATION MAP
1802
SYSTEM S-BLOCK INFORMATION TABLE
ENTRY 3 + XXXX
908
BUFFER
S-JOURNAL
S-JOURNAL
S-JOURNAL
S-JOURNAL
1004
2206
OPEN SYSTEM S-BLOCK 1
2208
SYSTEM S-BLOCK 3
S-JOURNAL
S-JOURNAL
S-JOURNAL
S-JOURNAL

FIG. 23

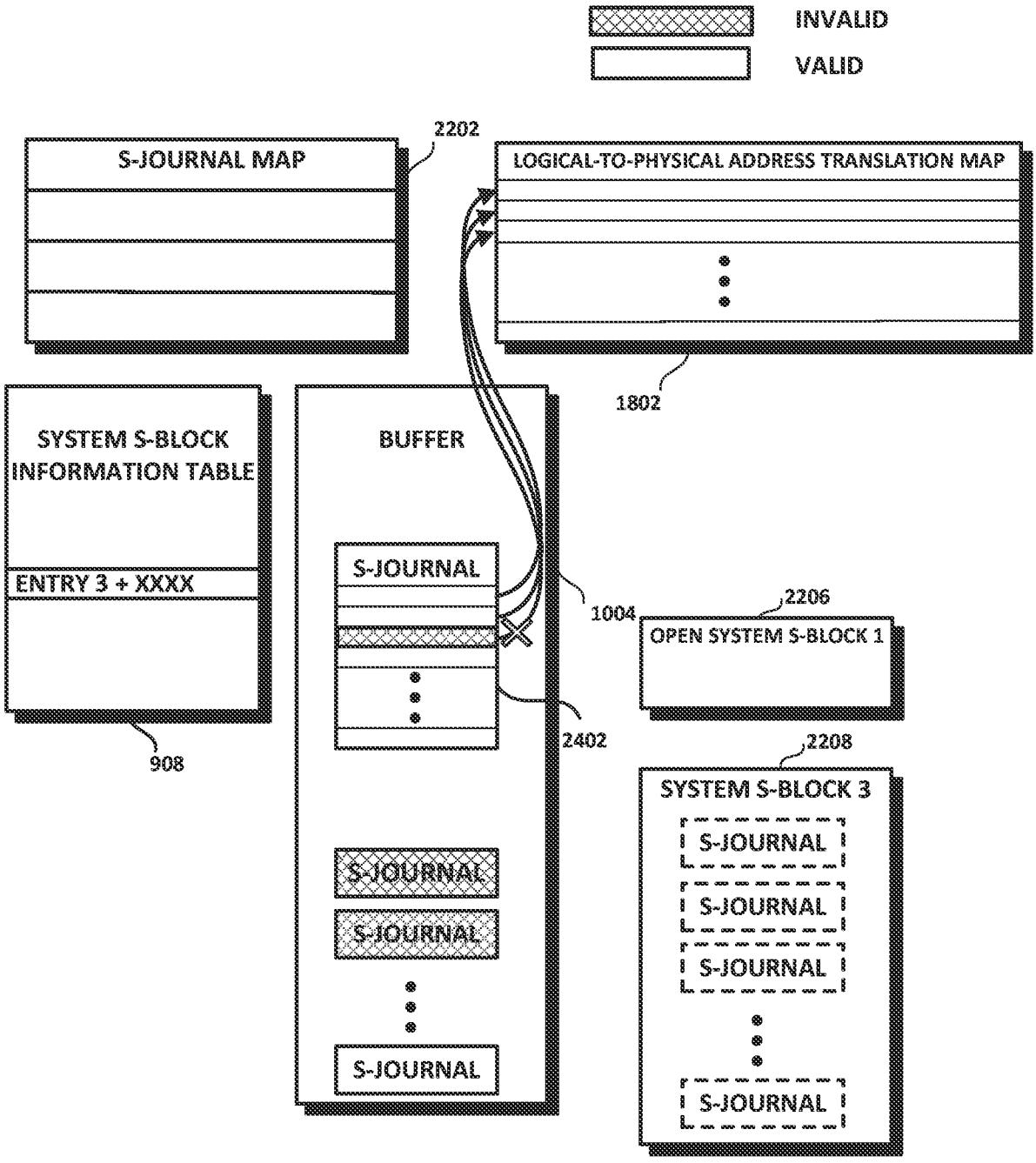

INVALID
VALID
S-JOURNAL MAP
2202
LOGICAL-TO-PHYSICAL ADDRESS TRANSLATION MAP
1802
SYSTEM S-BLOCK INFORMATION TABLE
ENTRY 3 + XXXX
908
BUFFER
S-JOURNAL
1004
2402
S-JOURNAL
S-JOURNAL
S-JOURNAL
2206
OPEN SYSTEM S-BLOCK 1
2208
SYSTEM S-BLOCK 3
S-JOURNAL
S-JOURNAL
S-JOURNAL
S-JOURNAL

FIG. 24

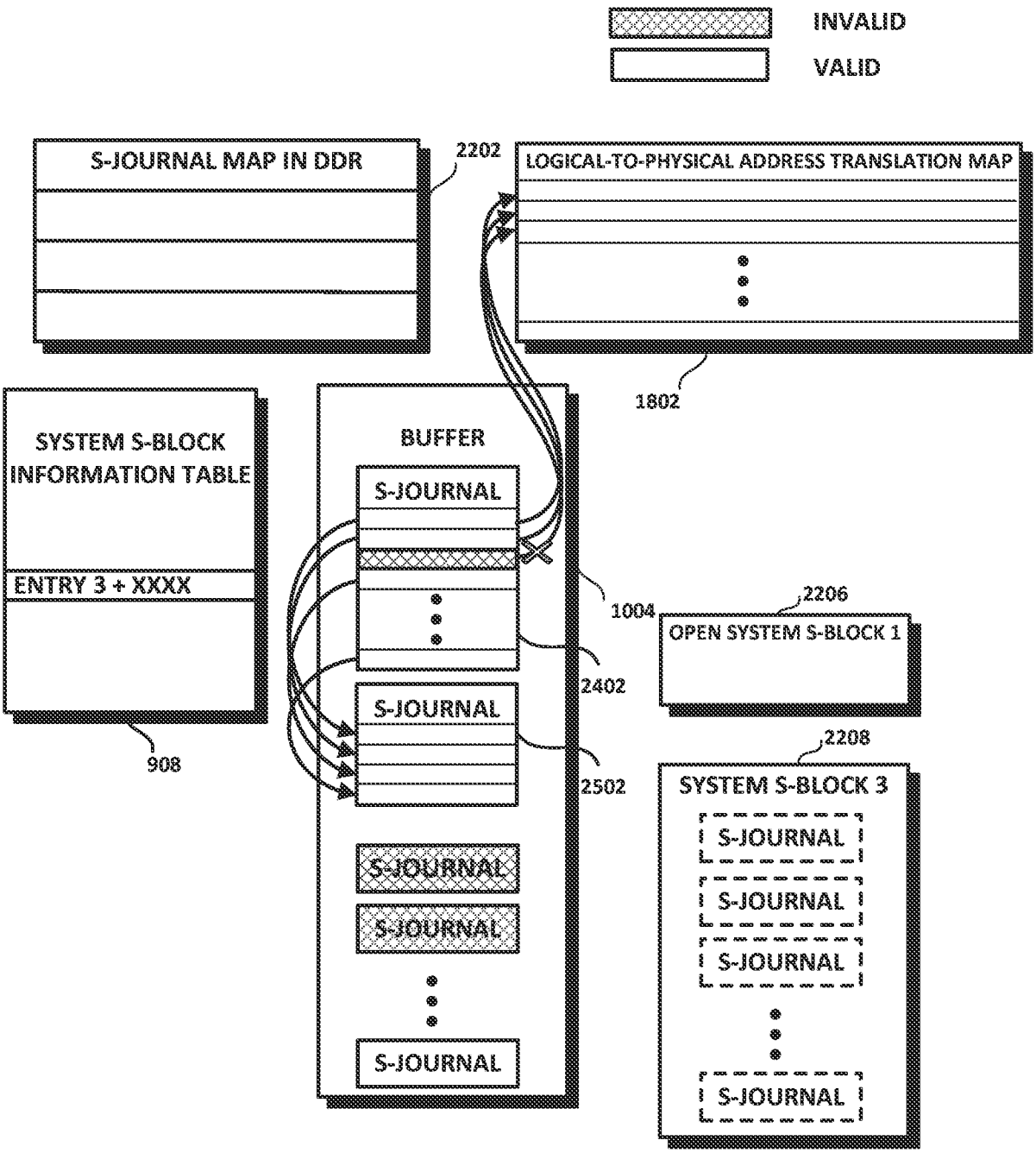

INVALID
VALID
S-JOURNAL MAP IN DDR
2202
LOGICAL-TO-PHYSICAL ADDRESS TRANSLATION MAP
1802
SYSTEM S-BLOCK INFORMATION TABLE
ENTRY 3 + XXXX
908
BUFFER
S-JOURNAL
1004
2402
S-JOURNAL
2502
S-JOURNAL
S-JOURNAL
S-JOURNAL
2206
OPEN SYSTEM S-BLOCK 1
2208
SYSTEM S-BLOCK 3
S-JOURNAL
S-JOURNAL
S-JOURNAL
S-JOURNAL

FIG. 25

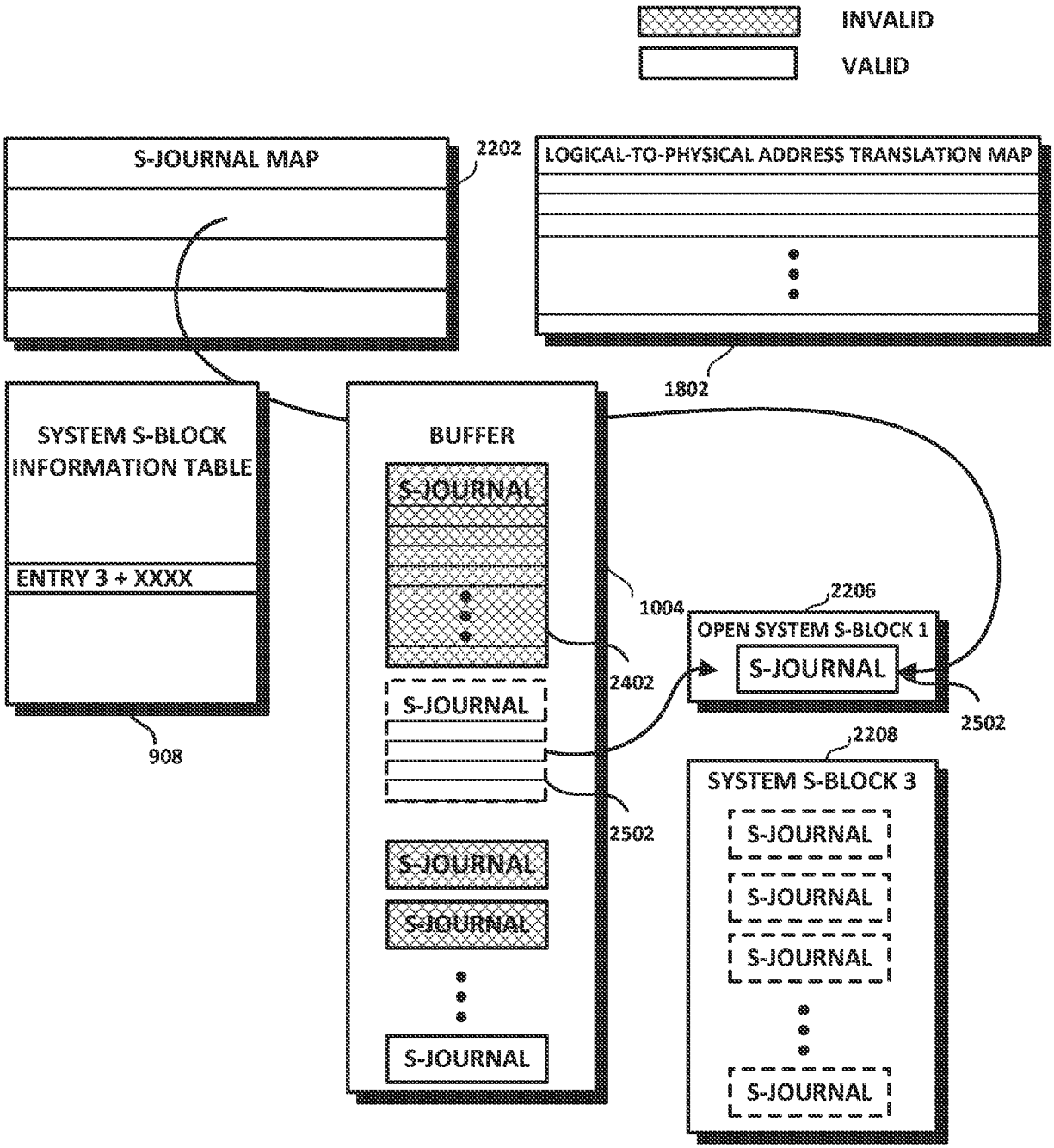

INVALID
VALID
S-JOURNAL MAP
2202
LOGICAL-TO-PHYSICAL ADDRESS TRANSLATION MAP
1802
SYSTEM S-BLOCK INFORMATION TABLE
ENTRY 3 + XXXX
908
BUFFER
S-JOURNAL
1004
2402
S-JOURNAL
2502
S-JOURNAL
S-JOURNAL
S-JOURNAL
2206
OPEN SYSTEM S-BLOCK 1
S-JOURNAL
2502
2208
SYSTEM S-BLOCK 3
S-JOURNAL
S-JOURNAL
S-JOURNAL
S-JOURNAL

FIG. 26

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2013/062723

A. CLASSIFICATION OF SUBJECT MATTER

G06F 12/00(2006.01)i

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 12/00; G11C 11/34; G06F 12/10; G06F 12/14

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords: solid state drives, logical, physical, mapping, volatile memory, non-volatile memory, page, address, journal, error correcting code,, and similar terms.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2010-0030999 A1 (TORSTEN HINZ) 04 February 2010 See paragraphs [0007] and [0020]-[0044]; claims 1-6; and figures 1-3.	1-45
A	US 2009-0049229 A1 (TOSHIYUKI HONDA et al.) 19 February 2009 See paragraphs [0011]-[0023] and [0051]-[0065]; claims 5 and 19-20; and figures 1-4.	1-45
A	US 2011-0072194 A1 (CARL FORHAN et al.) 24 March 2011 See paragraphs [0057]-[0066] and figures 1-4b.	1-45
A	US 2012-0226887 A1 (MARTIN L. CULLEY et al.) 06 September 2012 See paragraphs [0014]-[0027] and figure 1.	1-45
A	US 2012-0173795 A1 (FRANZ MICHAEL SCHUETTE et al.) 05 July 2012 See paragraphs [0038]-[0040] and figures 3-4.	1-45

☐ 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 application or patent 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
15 January 2014 (15.01.2014)	**16 January 2014 (16.01.2014)**

Name and mailing address of the ISA/KR
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701, Republic of Korea
Facsimile No. +82-42-472-7140

Authorized officer
NHO, Ji Myong
Telephone No. +82-42-481-8528

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2013/062723

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2010-0030999 A1	04/02/2010	DE 102009034651 A1	29/04/2010
		US 8321652 B2	27/11/2012
US 2009-0049229 A1	19/02/2009	CN 101176074 A	07/05/2008
		CN 101176074 B	15/12/2010
		EP 1895418 A1	05/03/2008
		EP 1895418 A4	27/08/2008
		JP 4633802 B2	23/02/2011
		US 8307149 B2	06/11/2012
		WO 2007-066720 A1	14/06/2007
US 2011-0072194 A1	24/03/2011	US 2011-0072162 A1	24/03/2011
		US 2011-0072173 A1	24/03/2011
		US 2011-0072187 A1	24/03/2011
		US 2011-0072196 A1	24/03/2011
		US 2011-0072197 A1	24/03/2011
		US 2011-0072198 A1	24/03/2011
		US 2011-0072199 A1	24/03/2011
		US 2011-0072209 A1	24/03/2011
		US 8219776 B2	10/07/2012
		US 8301861 B2	30/10/2012
		US 8312250 B2	13/11/2012
		US 8316178 B2	20/11/2012
		US 8352690 B2	08/01/2013
		US 8458381 B2	04/06/2013
		US 8504737 B2	06/08/2013
US 2012-0226887 A1	06/09/2012	TW 201250470 A	16/12/2012
		WO 2012-121968 A2	13/09/2012
		WO 2012-121968 A3	22/11/2012
US 2012-0173795 A1	05/07/2012	None	

(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 105027090 A
(43)申请公布日 2015.11.04

(21)申请号 201380063439.2
(22)申请日 2013.09.30
(30)优先权数据
13/645,822 2012.10.05 US
(85)PCT国际申请进入国家阶段日
2015.06.04
(86)PCT国际申请的申请数据
PCT/US2013/062723 2013.09.30
(87)PCT国际申请的公布数据
WO2014/055445 EN 2014.04.10
(71)申请人 西部数据技术公司
地址 美国加利福尼亚
申请人 天空时代有限责任公司
(72)发明人 A·J·汤姆林 J·琼斯
R·丹尼尔克 R·N·马伦多尔
(74)专利代理机构 永新专利商标代理有限公司 72002
代理人 刘瑜 王英
(51)Int.Cl.
G06F 12/00(2006.01)

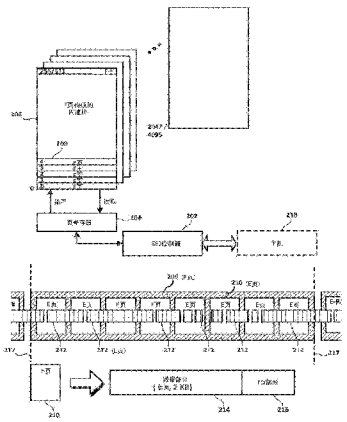
权利要求书5页 说明书12页 附图17页

(54)发明名称

用于固态驱动器中的物理到逻辑映射的方法、设备和系统

(57)摘要

一种数据存储设备，其包括存储物理页的多个非易失性存储器设备，所述物理页的每一个存储在预定的物理位置。控制器可以耦合到所述存储器设备，并且可以被配置为访问存储在多个逻辑页(L页)的数据，所述逻辑页的每一个与L页号相关联，使得所述控制器能够逻辑地引用存储在所述物理页中的数据。易失性存储器可以包括逻辑到物理地址转换映射，所述逻辑到物理地址转换映射使得所述控制器能够确定存储在每一个L页中的数据在所述物理页内的物理位置。所述控制器可以被配置为在所述存储器设备中保持限定了物理到逻辑对应关系的日志，每一个日志覆盖预定范围的物理页并且包括多个条目，所述多个条目将一个或多个物理页与每一个L页相关联。所述控制器可以在启动时读取所述日志，并且根据所读取的日志来重建所述地址转换映射。



CN 105027090 A

1. 一种数据存储设备，包括：

多个非易失性存储器设备，所述多个非易失性存储器设备的每一个被配置为存储多个物理页，所述多个物理页中的每一个存储在多个非易失性设备内预定的物理位置；

控制器，其耦合到多个存储器设备，并且被配置为将数据编程到所述多个存储器设备以及从所述多个存储器设备中读取数据，所述数据存储在多个逻辑页（L 页）中，所述多个 L 页中的每一个与 L 页号相关联，所述 L 页号被配置为使得所述控制器能够逻辑地引用存储在所述物理页的一个或多个中的数据；以及

易失性存储器，其包括逻辑到物理地址转换映射，所述逻辑到物理地址转换映射被配置为使得所述控制器能够确定存储在每一个 L 页中的数据在一个或多个物理页内的物理位置；

其中，所述控制器被配置为在所述多个非易失性存储器设备中保持限定了物理到逻辑对应关系的多个日志，所述多个日志中的每一个与日志号相关联，每一个日志覆盖预定范围的物理页并且包括多个日志条目，每一个条目被配置为将一个或多个物理页与每一个 L 页相关联，其中，所述控制器被配置为在启动时读取所述多个日志，并且根据所读取的多个日志对存储在所述易失性存储器中的地址转换映射进行重建。

2. 根据权利要求 1 所述的数据存储设备，其中，所述控制器进一步被配置为，在对所述多个 L 页中的一个进行更新时，在所述多个日志中的一个中创建新的条目。

3. 根据权利要求 2 所述的数据存储设备，其中，用于保持转换数据的电力安全拷贝的对所述非易失性存储器设备的写操作被配置为由新创建的日志条目而不是通过保存所述转换映射的至少一部分来触发，使得写放大被减少。

4. 根据权利要求 2 所述的数据存储设备，其中，所述新的条目指示所更新的 L 页的起始在物理页内的物理位置。

5. 根据权利要求 2 所述的数据存储设备，其中，所述控制器进一步被配置为将空闲空间更新为占有与所述 L 页被更新之前的长度相对应的量。

6. 根据权利要求 1 所述的数据存储设备，其中，所述多个 L 页中的至少一个没有与物理页边界对齐。

7. 根据权利要求 1 所述的数据存储设备，其中，所述物理页被实现为纠错码（ECC）页（E 页），并且其中，所述多个设备包括多个闪速存储器块，每一个闪速存储器块包括多个闪速存储器页（F 页），所述 F 页中的每一个包括多个 E 页，所述多个 E 页中的每一个存储在所述多个设备内的预定的物理位置处。

8. 根据权利要求 2 所述的数据存储设备，其中，所述控制器进一步被配置为利用由所更新的 L 页的 L 页号所引用的数据在一个或多个物理页内的物理位置来更新所述转换映射。

9. 根据权利要求 1 所述的数据存储设备，进一步包括多个超块（S 块），所述多个 S 块的每一个包括每设备的一个或多个闪速存储器块，并且其中，所述多个日志条目中的每一个被配置为将所述 S 块的物理页中的一个或多个与每一个 L 页相关联。

10. 根据权利要求 9 所述的数据存储设备，其中，所述控制器进一步被配置为通过至少以下操作来进行垃圾收集：

选择所述多个 S 块中的一个用于垃圾收集；

将针对所选择的 S 块的日志中的每一个条目与所述转换映射中的条目进行比较，并且将匹配的条目指定为有效的而将不匹配的条目指定为无效的；

读取与有效条目相对应的 L 页；

将所读取的 L 页写到所述多个非易失性存储器设备内的相应的物理地址；

对针对所述有效条目的转换映射进行更新，以使其指向所述相应的物理地址；以及

针对对其更新了转换映射的条目来生成新的日志条目。

11. 根据权利要求 9 所述的数据存储设备，其中，所述控制器进一步被配置为通过至少以下操作来进行垃圾收集：

选择所述多个 S 块中的一个来进行垃圾收集；

读取所选择的 S 块的物理页；

将所读取的所选择的 S 块的物理页中的 L 页号与所述转换映射中的条目进行比较，并且将匹配的条目指定为有效的而将不匹配的条目指定为无效的；

将与有效条目相对应的 L 页写到所述多个非易失性存储器设备内的相应的物理地址；

对针对所述有效条目的转换映射进行更新，以使其指向所述相应的物理地址；以及

针对对其更新了转换映射的条目来生成新的日志条目。

12. 根据权利要求 10 所述的数据存储设备，其中，选择包括在确定选择哪个 S 块中，对空闲空间和编程擦除 (PE) 计数进行加权。

13. 根据权利要求 1 所述的数据存储设备，其中，每一个日志号包括由所述日志覆盖的第一物理页的地址的预定数量的最高有效位。

14. 根据权利要求 1 所述的数据存储设备，其中，所述多个日志条目中的每一个包括：

L 页号，以及

物理地址位置。

15. 根据权利要求 1 所述的数据存储设备，其中，所述多个日志条目中的每一个包括：

L 页号；

物理页的物理地址位置，以及

L 页尺寸。

16. 根据权利要求 1 所述的数据存储设备，其中，所述多个日志条目中的每一个包括：

包括 L 页的起始的物理页的地址的预定数量的最低有效位；

地址；

L 页尺寸；以及

所述物理页内的偏移。

17. 根据权利要求 1 所述的数据存储设备，其中，所述多个 L 页被配置为被压缩并且在尺寸上不同，并且其中，所述多个日志号被配置为引用较大数量的较小尺寸的 L 页或者引用较少数量的较大尺寸的 L 页。

18. 根据权利要求 1 所述的数据存储设备，其中，所述控制器进一步被配置为在启动时以预定的序列顺序读取所述多个日志，并且基于顺序地读取的多个日志来对存储在易失性存储器中的所述转换映射进行重建。

19. 根据权利要求 1 所述的数据存储设备，其中，所述控制器进一步被配置为基于所述多个日志来建立日志映射。

20. 根据权利要求 1 所述的数据存储设备，其中，所述控制器进一步被配置为：

在启动时以预定的序列顺序读取所述多个日志；

基于顺序地读取的多个日志来建立存储在所述非易失性存储器设备中的日志的映射，以及

将所建立的日志的映射存储在所述易失性存储器中。

21. 根据权利要求 1 所述的数据存储设备，其中，在与给定的 L 页相关联的日志条目当中，仅与所述给定的 L 页相关联的最近更新的日志条目是有效的。

22. 根据权利要求 1 所述的数据存储设备，其中，所述控制器进一步被配置为在所述易失性存储器中保持所述多个日志的系统日志映射，所述系统日志映射中的每一个条目指向所述多个日志中的一个存储在所述非易失性存储器设备中的位置。

23. 一种控制包括易失性存储器和多个非易失性存储器设备在内的数据存储设备的方法，多个非易失性设备中的每一个被配置为存储多个物理页，所述多个物理页中的每一个存储在所述多个非易失性设备内预定的物理位置处，所述方法包括：

在多个逻辑页（L 页）中存储数据，所述多个 L 页中的每一个与 L 页号相关联，所述 L 页号被配置为使得控制器能够逻辑地引用存储在所述物理页的一个或多个中的数据；

在所述易失性存储器中保持逻辑到物理地址转换映射，所述转换映射被配置为使得能够确定存储在每一个 L 页中的数据在所述物理页的一个或多个内的物理位置；

在所述多个非易失性存储器设备中保持限定了物理到逻辑对应关系的多个日志，所述多个日志中的每一个与日志号相关联，每一个日志覆盖预定范围的物理页，并且每一个日志包括多个日志条目，每一个条目被配置为将一个或多个物理页与每一个 L 页相关联；以及

在启动时读取所述多个日志，并且基于所读取的所述多个日志中的条目来对存储在易失性存储器中的所述转换映射进行重建。

24. 根据权利要求 23 所述的方法，进一步包括在对所述多个 L 页中的一个进行更新时，在所述多个日志中的一个中创建新的条目。

25. 根据权利要求 24 所述的方法，进一步包括基于新创建的日志条目而不是保存所述转换映射的至少部分来触发用于保持转换数据的电力安全拷贝的对所述非易失性存储器设备的写操作，以使得写放大被减少。

26. 根据权利要求 24 所述的方法，其中，所述新的条目指示所更新的 L 页的起始在物理页内的物理位置。

27. 根据权利要求 24 所述的方法，进一步包括将空闲空间更新为占用与所述 L 页被更新之前的长度相对应的量。

28. 根据权利要求 23 所述的方法，其中，存储包括将所述多个 L 页的至少一个存储在不与物理页边界对齐的位置。

29. 根据权利要求 23 所述的方法，其中，所述物理页被实现为纠错码（ECC）页（E 页），并且其中，所述多个设备包括多个闪速存储器块，每一个闪速存储器块包括多个闪速存储器页（F 页），F 页中的每一个包括多个 E 页，所述多个 E 页中的每一个存储在所述多个设备内的预定的物理位置处。

30. 根据权利要求 23 所述的方法，进一步包括利用由所更新的 L 页的 L 页号所引用的

数据在一个或多个物理页内的物理位置来更新所述转换映射。

31. 根据权利要求 23 所述的方法，其中，所述多个设备包括多个超块 (S 块)，所述多个 S 块的每一个包括每设备的一个或多个闪速存储器块，并且其中，所述多个日志条目中的每一个被配置为将所述 S 块的物理页中的一个或多个与每一个 L 页相关联。

32. 根据权利要求 31 所述的方法，进一步包括 ：

选择 S 块用于垃圾收集 ；

将针对所选择的 S 块的日志中的每一个条目与所述转换映射中的条目进行比较，并且将匹配的条目指定为有效的而将不匹配的条目指定为无效的 ；

读取与有效条目相对应的 L 页 ；

将所读取的 L 页写到所述多个非易失性存储器设备内的相应的物理地址 ；

对针对所述有效条目的转换映射进行更新，以使其指向所述相应的物理地址 ；以及

针对对其更新了转换映射的条目来生成新的日志条目。

33. 根据权利要求 31 所述的方法，进一步包括 ：

选择所述多个 S 块中的一个用于垃圾收集 ；

读取所选择的 S 块的物理页 ；

将所读取的所选择的 S 块的物理页中的 L 页号与所述转换映射中的条目进行比较，并且将匹配的条目指定为有效的而将不匹配的条目指定为无效的 ；

将与有效条目相对应的 L 页写到所述多个非易失性存储器设备内的相应的物理地址 ；

对针对所述有效条目的转换映射进行更新，以使其指向所述相应的物理地址 ；以及

针对对其更新了转换映射的条目来生成新的日志条目。

34. 根据权利要求 32 所述的方法，其中，选择包括在确定选择哪个 S 块中，对空闲空间和编程擦除 (PE) 计数进行加权。

35. 根据权利要求 34 所述的方法，其中，所述日志号包括由所述日志覆盖的第一物理页的地址的预定数量的最高有效位。

36. 根据权利要求 23 所述的方法，其中，所述多个日志条目中的每一个包括 ：

L 页号，以及

物理地址位置。

37. 根据权利要求 23 所述的方法，其中，所述多个日志条目中的每一个包括 ：

L 页号 ；

物理页的物理地址位置，以及

L 页尺寸。

38. 根据权利要求 23 所述的方法，其中，所述多个日志条目中的每一个包括 ：

包括 L 页的起始的物理页的地址的预定数量的最低有效位 ；

地址 ；

L 页尺寸 ；以及

所述物理页内的偏移。

39. 根据权利要求 23 所述的方法，进一步包括选择性地对所述多个 L 页进行压缩，以使得所述多个 L 页在尺寸上不同，并且其中，所述多个日志被配置为引用较大数量的较小尺寸的 L 页或者引用较少数量的较大尺寸的 L 页。

40. 根据权利要求 23 所述的方法，其中，在启动时读取所述多个日志和重建所述转换映射包括：在启动时以预定的序列顺序来读取所述多个日志，并且基于所读取的多个日志来对存储在易失性存储器中的所述转换映射进行重建。

41. 根据权利要求 23 所述的方法，其中，所述控制器进一步被配置为基于所述多个日志来建立日志映射。

42. 根据权利要求 23 所述的方法，进一步包括：

在启动时以预定的序列顺序读取所述多个日志；

基于所顺序读取的多个日志来建立存储在所述非易失性存储器设备中的日志的映射，以及

将所建立的日志的映射存储在所述易失性存储器中。

43. 根据权利要求 23 所述的方法，其中，在与给定的 L 页相关联的日志条目当中，仅与所述给定的 L 页相关联的最新更新的日志条目是有效的。

44. 根据权利要求 23 所述的方法，进一步包括在所述易失性存储器中保持所述多个日志的系统日志映射，所述系统日志映射中的每一个条目指向所述多个日志中的一个存储在所述非易失性存储器设备中的位置。

45. 一种数据存储设备控制器，包括：

处理器，其被配置为耦合到易失性存储器和多个存储器设备，所述多个存储器设备中的每一个被配置为将多个物理页存储在多个非易失性设备内预定的物理位置处，所述处理器进一步被配置为将数据编程到所述多个存储器设备以及从所述多个存储器设备中读取数据，所述数据存储在多个逻辑页（L 页）中，所述多个 L 页中的每一个与 L 页号相关联，所述 L 页号被配置为使得所述处理器能够逻辑地引用存储在所述物理页的一个或多个中的数据，所述易失性存储器被配置为存储逻辑到物理地址转换映射，所述逻辑到物理地址转换映射被配置为使得所述处理器能够确定存储在每一个 L 页中的数据在一个或多个物理页内的物理位置，

其中，所述处理器被配置为在所述多个非易失性存储器设备中，保持限定了物理到逻辑对应关系的多个日志，所述多个日志中的每一个与日志号相关联，每一个日志覆盖预定范围的物理页并且包括多个日志条目，每一个条目被配置为将一个或多个物理页与每一个 L 页相关联，其中，所述处理器进一步被配置为在启动时读取所述多个日志，并且根据所读取的多个日志来对存储在所述易失性存储器中的地址转换映射进行重建。

用于固态驱动器中的物理到逻辑映射的方法、设备和系统

背景技术

[0001] 由于固态驱动器 (SSD) 中的闪速存储器的特性,数据通常是按页编程并且按块擦除的。SSD 中页的尺寸通常为 8-16 千字节 (KB) 并且块由很多的页 (例如,256 或 512) 组成。因此,SSD 中的特定物理位置 (例如,页) 不能在没有对同一块内的页中的数据进行重写的情况下直接进行重写,而这在磁性硬盘驱动器中是可能的。因此,需要间接地址。常规的数据存储设备控制器管理诸如 SSD 等的数据存储设备上的闪速存储器以及与主机系统进行连接,所述常规的数据存储设备控制器使用是闪存转换层 (FTL) 的一部分的、被称为逻辑块寻址 (LBA) 的逻辑到物理 (L2P) 映射系统。当新数据到达以替换已经写入的旧数据时,数据存储设备控制器使得新数据被写入在新的位置,并且更新逻辑映射以使其指向新的物理位置。由于旧的物理位置不再保存有效数据,所以在能够对旧的物理位置再次写之前最终需要将其擦除。

[0002] 常规地,大的 L2P 映射表将逻辑条目映射到 SSD 上的物理地址位置。大的 L2P 映射表可以驻留在诸如动态随机存取存储器 (DRAM) 等的易失性存储器中,所述大的 L2P 映射表通常随着写入被更新、到达、并且保存到小区域中的非易失性存储器。例如,如果随机写发生,尽管系统可能仅需要更新一个条目,但是系统可能不得不将包括没有更新的条目的整个表或其一部分保存到非易失性存储器,这在本质上是低效的。

[0003] 图 1 示出了用于 SSD 的常规的逻辑块寻址 (LBA) 方案的方面。如其中所示,映射表 104 包含针对数据存储设备的闪速存储器 106 限定的针对每个逻辑块 102 的一个条目。例如,支持 512 字节逻辑块的 64 GB SSD,自身可以向主机呈现为如同具有 125000000 个逻辑块。在映射表 104 中的一个条目包含在闪速存储器 106 中的 125000000 个逻辑块的每一个逻辑块的当前位置。在常规 SSD 中,闪速页容纳整数个逻辑块 (即,逻辑块不跨越闪速页)。在该常规示例中,8KB 的闪速页将容纳 16 个 (尺寸为 512 字节的) 逻辑块。因此,在逻辑到物理映射表 104 中的每一个条目包含 :字段 108,其用于标识在其上存储逻辑块的闪速管芯,字段 110,其用于标识在其上存储逻辑块的闪速块,另一字段 112,其用于标识闪速块内的闪速页,以及字段 114,其用于标识闪速页内的偏移,该偏移标识逻辑块数据在所标识的闪速页中的何处开始。大尺寸的映射表 104 使得表不能保存在 SSD 控制器的内部。常规地,将该大的映射表 104 保存在连接到 SSD 控制器的外部 DRAM 中。因为将映射表 104 存储在易失性DRAM中,所以当SSD上电时必须恢复映射表104,这由于表的大尺寸可能耗费很长的时间。

[0004] 当读取逻辑块时,读取映射表 104 中对应的条目以确定要被读取的闪速存储器中的位置。然后向映射表 104 中对应的条目中所指定的闪存页执行读取。当读取的数据对于该闪存页可用时,在由映射条目所指定的偏移处的数据从 SSD 传送到主机。当写逻辑块时,对映射表 104 中对应的条目进行更新以反映该逻辑块的新位置。应当注意的是,当写逻辑块时,闪速存储器初始地将包含至少两个版本的逻辑块 ;即,有效的、最近写入的版本 (由映射表 104 所指向的) 和其至少一个另外的、较旧的版本,该较旧版本是陈旧的并且不再由映射表 104 中的任意条目所指向。这些“陈旧的”数据被称为垃圾,这些垃圾占用的空间必

须被记录、收集、擦除并使其以供未来使用。

附图说明

[0005] 图 1 示出了用于 SSD 的常规的逻辑块寻址 (LBA) 方案的方面。

[0006] 图 2 是示出了根据一个实施例的数据存储设备的物理数据组织结构和逻辑数据组织结构的方面的图。

[0007] 图 3 示出了根据一个实施例的逻辑到物理地址转换映射以及其示例性条目。

[0008] 图 4 示出了根据一个实施例的用于更新逻辑到物理地址转换映射以及用于创建 S 日志条目的方法的方面。

[0009] 图 5 是根据一个实施例的 S 日志的框图。

[0010] 图 6 示出了根据一个实施例的 S 日志的一个条目的示例性组织结构。

[0011] 图 7 是根据一个实施例的超块 (S 块) 的框图。

[0012] 图 8 示出了根据一个实施例的超页 (S 页) 的另一视图。

[0013] 图 9A 示出了根据一个实施例的逻辑到物理地址转换映射、S 日志和 S 块之间的关系。

[0014] 图 9B 是根据一个实施例的 S 日志映射的框图。

[0015] 图 10 是示出了根据一个实施例的对逻辑到物理地址转换映射进行更新的方法的方面的框图。

[0016] 图 11 是示出了根据一个实施例的对逻辑到物理地址转换映射进行更新的方法的另一方面的框图。

[0017] 图 12 是示出了根据一个实施例的对逻辑到物理地址转换映射进行更新的方法的又一方面的框图。

[0018] 图 13 是示出了根据一个实施例的对逻辑到物理地址转换映射进行更新的方法的还一方面的框图。

[0019] 图 14 是示出了根据一个实施例的对 S 日志和 S 日志映射进行更新的方法的方面的框图。

[0020] 图 15 是示出了根据一个实施例的对 S 日志和 S 日志映射进行更新的方法的另一方面的框图。

[0021] 图 16 是示出了根据一个实施例的对 S 日志和 S 日志映射进行更新的方法的又一方面的框图。

[0022] 图 17 是示出了根据一个实施例的对 S 日志和 S 日志映射进行更新的方法的还一方面的框图。

[0023] 图 18 是示出了根据一个实施例的垃圾收集的方面的框图。

[0024] 图 19 是示出了根据一个实施例的垃圾收集的另一方面的框图。

[0025] 图 20 是示出了根据一个实施例的垃圾收集的又一方面的框图。

[0026] 图 21 是示出了根据一个实施例的垃圾收集的还一方面的框图。

[0027] 图 22 是示出了根据一个实施例的对系统块进行垃圾收集的方面的框图。

[0028] 图 23 是示出了根据一个实施例的对系统块进行垃圾收集的另一方面的框图。

[0029] 图 24 是示出了根据一个实施例的对系统块进行垃圾收集的又一方面的框图。

[0030] 图 25 是示出了根据一个实施例的对系统块进行垃圾收集的其他方面的框图。

[0031] 图 26 是示出了根据一个实施例的对系统块进行垃圾收集的又其他方面的框图。

具体实施方式

[0032] 系统概述

[0033] 图 2 是示出了根据一个实施例的数据存储设备的物理数据组织结构和逻辑数据组织结构的方面的图。在一个实施例中，数据存储设备是 SSD。在另一实施例中，数据存储设备是包括闪速存储器和旋转磁存储介质在内的混合驱动器。本公开能够应用于 SSD 和混合实现二者，但是出于简单的缘故，参考基于 SSD 的实现来对各种实施例进行描述。根据一个实施例，数据存储设备控制器 202 可以被配置为耦合到如附图标记 218 处所示的主机。主机 218 可以采用逻辑块寻址 (LBA) 方案。尽管 LBA 的尺寸通常是固定的，但是主机能够动态地改变 LBA 的尺寸。例如，LBA 的尺寸可以随着接口和接口模式而不同。实际上，尽管 512 字节是最常见的，但 4KB 也变得更加常见，512+(520、528 等) 和 4KB+(4KB+8、4KB+16 等) 格式同样常见。如其中所示，数据存储设备控制器 202 可以包括页寄存器 204 或耦合到页寄存器 204。页寄存器 204 可以被配置为使控制器 202 能够从数据存储设备读取数据以及将数据存储到数据存储设备。控制器 202 可以被配置为响应于来自主机 218 的数据访问命令而进行编程并且从闪速存储器设备的阵列读取数据。尽管本文的描述通常涉及闪速存储器，但是应当理解的是，存储器设备的阵列可以包括各种类型的非易失性存储器设备中的一个或多个，所述非易失性存储器设备例如是，闪速集成电路、硫系 RAM(C-RAM)、相变存储器 (PC-RAM 或 PRAM)、可编程金属化单元 RAM(PMC-RAM 或 PMCm)、标准化存储器 (OUM)、阻变式 RAM(RRAM)、NAND 存储器 (例如，单层单元 (SLC) 存储器、多层单元 (MLC) 存储器或其任意组合)、NOR 存储器、EEPROM、铁电体存储器 (FeRAM)、磁阻 RAM(MRAM)、其他分立的 NVM(非易失性存储器) 芯片、或其任意组合。

[0034] 页寄存器 204 可以被配置为使控制器 202 能够从阵列中读取数据以及将数据存储到阵列。根据一个实施例，闪速存储器设备的阵列可以包括管芯 (例如，128 个管芯) 中的多个非易失性存储器设备，其中的每一个包括多个块，如图 2 中的 206 处所示。其他页寄存器 204(未示出) 可以耦合到其他管芯上的块。集合在一起的闪速块的组合可以被称为超块或 S 块。在一些实施例中，形成 S 块的单独的块可以从一个或多个管芯、平面或其他粒度级别中选择。因此，S 块可以包括组合在一起的、分布跨过一个或多个管芯的多个闪速块。以这种方式，S 块可以形成闪速管理系统 (FMS) 在其上操作的单元。在一些实施例中，可以根据与在管芯级不同的粒度来选择形成 S 块的单独的块，例如以下情况 :当存储器设备包括被细分为诸如平面等的结构的管芯 (即，块可以从单独的平面中取得) 时。根据一个实施例，分配、擦除和垃圾收集可以在 S 块级实施。在其他实施例中，FMS 可以根据诸如页、块、平面、管芯等的其他逻辑分组来执行数据操作。

[0035] 继而，每一个闪速块 206 包括多个闪速页 (F 页)208。每一个 F 页可以是固定尺寸的，例如，16KB。根据一个实施例，F 页是用于给定的闪速设备的编程的最小单元的尺寸。同样如图 2 所示，每一个 F 页 208 可以被配置为容纳多个物理页，在下文中被称为 E 页 210。术语“E 页”指的是存储在其上应用了纠错码 (ECC) 的闪速存储器中的数据结构。根据一个实施例，E 页 210 可以形成数据存储设备中的物理寻址的基础，并且可以构成闪速读取数

据传送的最小单元。因此，E 页 210 可以具有（但不是必须具有）预定的固定尺寸（例如，2KB），并且确定 ECC 系统的有效载荷（例如，主机数据）的尺寸。根据一个实施例，每一个 F 页 208 可以被配置为使预定的多个 E 页 210 适合在其边界内。例如，给定 16KB 尺寸的 F 页 208 以及每 E 页 210 固定尺寸为 2KB，八个 E 页 210 适合在单个 F 页 208 内，如图 2 所示。根据一个实施例，无论如何，2 的幂次乘以包括 ECC 的 E 页 210 可以被配置为适合于 F 页 208。每一个 E 页 210 可以包括数据部分 214，并且取决于 E 页 210 位于何处也可以包括 ECC 部分 216。数据部分 214 和 ECC 部分 216 在尺寸上都不需要是固定的。E 页的地址唯一地标识在闪速存储器内 E 页的位置。例如，E 页的地址可以指定闪速通道、在所标识的闪速通道内的特定管芯、在管芯内的特定块、特定 F 页以及最后在所标识的 F 页中的 E 页。

[0036] 为了在数据存储设备上的物理寻址与由主机进行的逻辑块寻址之间建立桥梁，引入了逻辑页（L 页）结构。图 2 中附图标记 212 处所指示的 L 页可以包括由 FMS 所使用的地址转换的最小单元。根据一个实施例，每一个 L 页可以与 L 页号相关联。因此，L 页 212 的 L 页号可以被配置为使控制器 202 能够对存储在诸如 E 页 210 等的物理页的一个或多个中的主机数据进行逻辑地引用。也可以利用 L 页 212 作为压缩的基本单元。根据一个实施例，与 F 页 208 和 E 页 210 不同，L 页 212 的尺寸是不固定的，并且由于对要存储的数据进行压缩中的可变性，L 页 212 在尺寸上可以不同。由于数据的可压缩性不同，所以例如 4KB 量的一种类型的数据可以被压缩成 2KB 的 L 页，而 4KB 量的另一种类型的数据可以被压缩成 1KB 的 L 页。因此，由于这样的压缩，L 页的尺寸可以在一个范围内变化，所述范围由例如 24 字节的最小压缩尺寸到例如 4KB 或 4KB+ 的最大解压缩尺寸进行限定。可以实现其他尺寸和范围。如图 2 所示，L 页 212 不需要与 E 页 210 的边界对齐。实际上，L 页 212 可以被配置为具有与 F 页 208 和 / 或 E 页 210 边界对齐的起始地址，但也可以被配置为不与 F 页 208 或 E 页 210 的边界对齐。即，L 页起始地址所处的位置可以与 F 页 208 的起始地址或终止地址相距非零偏移或者与 E 页 210 的起始地址或终止地址相距非零偏移，如图 2 所示。由于 L 页 212 在尺寸上是不固定的并且可能小于固定尺寸的 E 页 210，所以多于一个的 L 页 212 可以适合在单个 E 页 210 中。同样地，由于 L 页 212 在尺寸上可能大于 E 页 210，所以 L 页 212 可能跨越多于一个 E 页并且甚至可能跨过 F 页 210 的边界，如图 2 中附图标记 217 处所示。

[0037] 例如，在 LBA 尺寸是 512 或 512+ 字节的情况，假定解压缩的 L 页 212 可以是 4KB 到 4KB+，则例如最大八个连续的 LBA 可以被紧缩为 4KB 的 L 页 212。根据一个实施例，应当注意的是，L 页 212 的确切逻辑尺寸不重要，这是因为在压缩之后物理尺寸的范围可以从最小尺寸若干字节跨越到完整尺寸上千字节。例如，对于 4TB 的 SSD 设备，可以使用 30 位地址来对每一个 L 页 212 进行编址以覆盖可能潜在地呈现在这样的 SSD 中的大量的 L 页。

[0038] 图 3 示出了根据一个实施例的逻辑到物理地址转换映射以及其示例性条目。由于主机数据在 L 页 212 中由主机引用，并且由于数据存储设备将 L 页 212 存储在一个或多个邻接的 E 页 210 中，所以需要逻辑到物理地址转换映射以使控制器 202 能够将 L 页 212 的 L 页号与一个或多个 E 页 210 相关联。这样的逻辑到物理地址转换映射在图 3 中 302 处示出，并且在一个实施例中，逻辑到物理地址转换映射是每 L 页 212 具有一个条目的线性阵列。这样的逻辑到物理地址转换映射 302 可以存储在诸如 DRAM 或 SRAM 等的易失性存储器中。图 3 还示出了用于四个不同的 L 页 212 的逻辑到物理地址转换映射中的条目，其中，图

3 中的 L 页 212 与被指示为 L 页 1、L 页 2、L 页 3 和 L 页 4 的 L 页号相关联。根据一个实施例，存储在数据存储设备中的每一个 L 页可以由逻辑到物理地址转换映射 302 中的单个并且唯一的条目所指向。因此，在以此产生的示例中，示出了四个条目。如 302 处所示，映射 302 中的每一个条目可以包括 L 页号，所述 L 页号可以包括物理页（例如，E 页）的标识，所述 L 页号包含被引用的 L 页的起始地址、在物理页（例如，E 页）内对起始地址的偏移和 L 页的长。另外，多个 ECC 位可以针对映射条目来提供错误纠正功能。例如，以及如图 3 所示，并且假设 E 页尺寸为 2KB，L 页 1 可以在逻辑到物理地址转换映射 302 中被引用如下：E 页 1003、偏移 800、长 1624，紧接着是预定数量的 ECC 位（未示出）。即，在物理地址项中，L 页 1 的起始是在 E 页 1003 内（不与 E 页 1003 对齐），并且所处的位置与 E 页 1003 的起始物理位置相距等于 800 字节的偏移。而且，压缩的 L 页 1 扩展 1624 字节，从而跨过 E 页边界到达 E 页 1004。因此，E 页 1003 和 E 页 1004 每一个都存储了由 L 页号 L 页 1 所指示的 L 页 212 的一部分。类似地，由 L 页号 L 页 2 所引用的压缩的 L 页全部存储在 E 页 1004 内，并且在其中的偏移为 400 字节处起始，并仅在 E 页 1004 内扩展 696 字节。与 L 页号 L 页 3 相关联的压缩的 L 页起始于 E 页 1004 内偏移 1120 字节处（距离 L 页 2 的边界正好 24 字节），并且扩展 4096 字节越过 E 页 1005 并到达 E 页 1006 内。因此，与 L 页号 L 页 3 相关联的 L 页跨越 E 页 1004 的一部分、E 页 1005 的全部以及 E 页 1006 的一部分。最后，与 L 页号 L 页 4 相关联的 L 页开始于 E 页 1006 内的偏移 1144 字节处，并且扩展 3128 字节以完全跨越 E 页 1007、跨过 F 页边界到达 E 页 1008 的下一个 F 页中。

[0039] 共同地，组成了逻辑到物理地址转换映射 302 的每个条目的这些构成标识字段 (E 页、偏移、长度和 ECC) 中的每一个在尺寸上可以是诸如 8 字节。即，对于示例性 4TB 的驱动器，E 页的地址在尺寸上可以是 32 位，偏移在尺寸上可以是 12 位（对于 E 页数据部分多达 4KB），长度在尺寸上可以是 10 位，并且可以提供 ECC 字段。其他组织结构和位宽是可能的。每当写入或改变 L 页时，可以创建这样的 8 字节条目，使得控制器 202 能够跟踪在闪速存储装置内写入到 L 页的主机数据。在逻辑到物理地址转换映射中的所述 8 字节条目可以由 L 页号 LPN 来进行索引。换言之，根据一个实施例，L 页号充当逻辑到物理地址转换映射 302 的索引。应当注意的是，在 4KB 扇区尺寸的情况下，LBA 与 LPN 相同。因此，LPN 可以构成易失性存储器内的条目的地址。当控制器 202 从主机 218 接收到读取命令时，LPN 可以从所供应的 LBA 中得到并且用于对逻辑到物理地址转换映射 302 进行索引以提取闪速存储器中要被读取的数据的位置。当控制器 202 从主机接收到写命令时，LPN 可以利用 LBA 进行构建并且可以对逻辑到物理地址转换映射 302 进行修改。例如，可以在其中创建新的条目。取决于存储逻辑到物理地址转换映射 302 的易失性存储器的尺寸，LPN 可以存储在单个条目中或被拆分为：例如，第一条目，其标识包含正讨论的 L 页的起始地址的 E 页（加 ECC 位）；以及第二条目，其标识偏移和长度（加 ECC 位）。因此，根据一个实施例，这两个条目可以一起与闪速存储器内的单个 L 页相对应并且指向闪速存储器内的单个 L 页。在其他实施例中，逻辑到物理地址转换映射条目的特定格式可以与以上所示示例不同。

[0040] S 日志与 S 日志映射

[0041] 由于逻辑到物理地址转换映射 302 可以存储在易失性存储器中，所以可能需要在启动或在对易失性存储器的任何其他电力损失时对逻辑到物理地址转换映射 302 进行重建。因此，这需要将一些机制和信息存储在非易失性存储器中，所述机制和信息将使得控制

器 202 能够在启动之后或断电事件之后“知道”L 页存储在非易失性存储器的何处之前重构逻辑到物理地址转换映射 302。根据一个实施例，将这样的机制和信息体现在可以称为系统日志或 S 日志的结构中。根据一个实施例，控制器 202 可以被配置为在多个非易失性存储器设备（例如，在一个或多个管芯、通道或平面的块 206 的一个或多个中）中，保持限定了物理到逻辑地址对应性的多个 S 日志。根据一个实施例，每一个 S 日志覆盖预定范围的物理页（例如，E 页）。根据一个实施例，每一个 S 日志可以包括多个日志条目，其中每一个条目被配置为将一个或多个物理页（例如，E 页）关联到每一个 L 页的 L 页号。根据一个实施例，每当控制器 202 重启时或无论何时逻辑到物理地址转换映射 302 被部分地或全部地重建时，控制器 202 读取 S 日志，并且根据从 S 日志条目读取的信息来重建逻辑到物理地址转换映射 302。

[0042] 图 4 示出了根据一个实施例的用于更新逻辑到物理地址转换映射以及用于创建 S 日志条目的方法的方面。如其中所示，为了保证使逻辑到物理地址转换映射 302 被保持为最新，无论何时 L 页被写入或另外被更新，如框 B41 所示，逻辑到物理地址转换映射 302 可以被更新，如 B42 所示。如 B43 所示，还可以创建 S 日志条目，所述 S 日志条目在其中存储指向更新的 L 页的位置的信息。以这种方式，当新的写发生时（例如，当主机向非易失性存储器发出写时、当垃圾收集 / 耗损均衡发生时等），逻辑到物理地址转换映射 302 和 S 日志二者都被更新。因此，用于保持地址转换数据的电力安全拷贝的向非易失性存储器设备的写操作可以被配置为由新创建的日志条目（其在尺寸上可能只有若干字节）所触发，而不是重新保存逻辑到物理地址转换映射的全部或一部分，这使得写放大 (WA) 被减小。对 S 日志的更新保证了控制器 202 能够访问新更新的 L 页，并且保证了逻辑到物理地址转换映射 302 可以在重启或影响逻辑到物理地址转换映射存储于其上的易失性存储器的其他擦除信息的电力事件之后被重建。此外，除了其在重建逻辑到物理地址转换映射 302 中的应用之外，S 日志在使垃圾收集 (GC) 能够高效中是有用的。实际上，S 日志可以包含对所有 L 页号的最新更新，并且还可以包含陈旧条目、不指向有效 L 页的条目。

[0043] 根据一个实施例，S 日志可以是写到非易失性存储器的主要的闪速管理数据。S 日志可以包含给定 S 块的映射信息，并且可以包含给定 S 块的物理到逻辑 (P2L) 信息。图 5 是示出了根据一个实施例的 S 日志的方面的框图。如其中所示，每一个 S 日志 502 覆盖非易失性存储器的预定的物理区域，例如，如 506 处所示的使用 5 位可编址的 32 个 E 页。每一个 S 日志 502 可以由 S 日志号进行标识，S 日志号可以是标头 504 的部分，标头 504 可以包括与 S 日志有关的其他信息。S 日志号可以包括由 S 日志所覆盖的第一物理页的地址的一部分。例如，S 日志 502 的 S 日志号可以包括 :例如，由所述 S 日志 502 所覆盖的第一 E 页地址的 27 个最高有效位 (MSb)。

[0044] 图 6 示出了根据一个实施例的 S 日志 502 的一个条目 602 的示例性组织结构。S 日志 502 的每一个条目 602 可以指向一个 L 页的起始地址，该 L 页的起始地址在物理上定址在 E 页中。每一个条目 602 可以包括例如包含起始 L 页的 E 页的地址的多个（例如，5）最低有效位 (LSb)。通过将所述 5 个 LSb 与标头 504 中的 S 日志号的 27 个 MSb 连接来获得全 E 页地址。另外，条目 602 可以包括 L 页号、其在所标识的 E 页内的偏移和其尺寸。例如，S 日志的每一个条目 602 可以包括由该 S 日志条目所覆盖的第一 E 页的地址的 5 个 LSb、30 位的 L 页号、9 位的 E 页偏移和 10 位的 L 页尺寸，合计达总尺寸约 7 字节。在其他实施例

中，可以使用各种其他内部日志条目格式。

[0045] 根据一个实施例，由于压缩中的可变性或存储在 L 页中的数据的主机配置，可以将可变数量的 L 页存储在物理区域（例如，等于 32 个 E 页的物理区域）中，如 506 处所示。由于压缩的使用和随之产生的 L 页的尺寸上的可变性，S 日志可以包括可变数量的条目。例如，根据一个实施例，在最大压缩的情况下，L 页在尺寸上可以是 24 字节，并且 S 日志可以包括超过 2500 个条目，其引用相等数量的 L 页，每 S 日志条目 602 一个 L 页。

[0046] 如上所述，S 日志可以被配置为包含给定 S 块的映射信息。更精确地，根据一个实施例，S 日志包含针对给定 S 块内的预定范围的 E 页的映射信息。图 7 是根据一个实施例的 S 块的框图。如其中所示，S 块 702 可以包括每管芯一个闪速块 (F 块)704（也如图 2 中的 206 处所示）。因此，可以把 S 块看作 F 块的集合，每管芯一个 F 块，所述 F 块被组合在一起以形成闪速管理系统的单元。根据一个实施例，分配、擦除和 GC 可以在 S 块级管理。如图 7 所示，每一个 F 块 704 可以包括多个闪速页 (F 页)，例如，256 或 512 个 F 页。根据一个实施例，F 页可以是用于给定的非易失性存储器设备的编程的最小单元的尺寸。图 8 示出了根据一个实施例的超页 (S 页)。如其中所示，S 页 802 可以包括每 S 块中的 F 块中的一个 F 页，这意味着 S 页跨越整个 S 块。

[0047] 各种数据结构之间的关系

[0048] 图 9A 示出了根据一个实施例的逻辑到物理地址转换映射、S 日志映射和 S 块之间的关系。附图标记 902 指示逻辑到物理地址转换映射中的条目（在一个实施例中存储在 DRAM 中）。根据一个实施例，逻辑到物理地址转换映射可以由 L 页号来进行索引，这是因为在逻辑到物理地址转换映射中每 L 页可以有一个条目 902。闪速存储器中的 L 页的起始的物理地址以及其尺寸可以在映射条目 902 中给出；即以 E 页地址，E 页内的偏移和 L 页的尺寸的方式。如早先所述，取决于 L 页的尺寸，L 页可以跨越一个或多个 E 页并且也可以跨越 F 页和 F 块。

[0049] 如 904 处所示，易失性存储器（例如，DRAM）也可以存储系统日志 (S 日志）映射。在 S 日志映射中的条目 904 存储关于 S 日志物理地位于非易失性存储器中何处的信息。例如，存储 L 页的起始位置的 E 页的物理地址的 27 个 MSb 可以构成 S 日志号（如先前图 5 中所示）。易失性存储器中的 S 日志映射条目 904 也可以包括在系统 E 页中引用的、在非易失性存储器中的 S 日志的地址。可以从易失性存储器中的 S 日志映射条目 904 中提取系统 S 块信息 908。系统 S 块信息 908 可以由系统 S 块（在系统带中的 S 块）进行索引，并且可以包括系统 S 块中任何空闲或被使用的空间的尺寸，连同关于 S 块的其他信息一起。同样从 S 日志映射条目 904 中，可以提取在非易失性存储器 910 中所引用的 S 日志的物理位置（按照系统带中的 E 页来表示）。

[0050] 根据一个实施例，系统带不包含 L 页数据，而可以包含文件管理系统 (FMS) 元数据和信息。系统带可以被配置为仅用于可靠性和断电简化的较低页。在正常操作期间，系统带不需要被读取，除非在垃圾收集期间。系统带可以被提供有比数据带显著较高的过剩供应以用于整体的 WA 优化。其他带包括：可以包含 L 页数据并且被频繁更新的热带，以及可以较少频繁更新并且可以包括较多静态数据的冷带，所述静态数据例如可以被收集作为 GC 的结果的数据。根据一个实施例，系统带、热带和冷带可以由 S 块基来进行分配。

[0051] 如上所述，在非易失性存储器中的这些 S 日志中的每一个可以包括 S 日志条目的

集合，并且覆盖例如价值 32 个 E 页的数据。非易失性存储器 910 中的这些 S 日志使得控制器 202 不仅能够重建易失性存储器中的逻辑到物理地址转换映射，而且能够重建易失性存储器中的 S 日志映射、用户 S 块信息 906 和系统 S 块信息 908。

[0052] 图 9B 是根据一个实施例的 S 日志映射 912 的框图。S 日志映射 912 可以由 S 块号来进行索引，并且其每一个条目可以指向该 S 块的第一 S 日志的起始位置，其继而可以覆盖该 S 块的预定数量的 E 页（例如，32 个）。控制器 202 可以进一步被配置为建立或重建 S 日志的映射，以及将结果的 S 日志映射存储在易失性存储器中。即，在重启时或者在另一个断电事件发生时或者错误修复随后的重启之后，控制器 202 可以以预定的序列顺序来读取多个 S 日志，基于顺序读取的多个 S 日志来建立存储在非易失性存储器设备中的 S 日志的映射，并且将所建立的 S 日志映射 912 存储在易失性存储器中。

[0053] 更新逻辑到物理地址转换映射

[0054] 图 10-13 是示出了根据一个实施例的对逻辑到物理地址转换映射进行更新的方法的方面的框图。如图 10 所示，逻辑到物理地址转换映射 1002 包含 L 页 100 的条目（例如，L 页的位置），所述条目具有 3012 字节的长度。在该示例中，L 页 100 存储在 S 块 15 中，如 1006 处所示。在控制器 202 中的或耦合到控制器 202 的缓冲器 1004（例如，静态随机存取存储器 (SRAM)）可以存储包含 L 页 100 驻留于其中的 S 块 15 的 P2L 信息的 S 日志。在一些实施例中，缓冲器 1004 中所示内容实际上可以驻留于 DRAM 内。其条目由 S 块进行索引的用户 S 块信息 906 对于每个 S 块而言，可以包括该 S 块中的空闲或使用的空间（例如，精确的或近似的）尺寸，连同关于 S 块的其他信息一起。如图 10 所示，示出了针对 S 块 15 的用户 S 块信息 906 中的条目。图 10 示出了在控制器 202 处理对 L 页 100 的更新之前，这些组成的功能块的示例性状态。

[0055] 如图 11 的 1102 处所示，接收到更新的 L 页 100，新长度为 1534 字节。响应于接收到更新的 L 页 100，L 页 100 的信息可以从逻辑到物理地址转换映射 1002 中取得，并且（现在已废弃的）L 页 100 的长度 (3012 字节）可以从逻辑到物理地址转换映射 1002 中提取，并用于相应地和确切地增加所跟踪的 S 块 15 的空闲空间值。特别地，用户 S 块信息 906 可以利用以下数据进行更新，所述数据指示 :S 块 15 现在具有 3012 额外字节的空闲空间，现在 L 页 100 的原始数据已经陈旧（例如，废弃）。

[0056] 如图 12 所示，可以对逻辑到物理地址转换映射 1002 进行更新以容纳所更新的 L 页。例如，长度信息现在是 1534 字节。其后，现在可以利用包括长度信息的新 L 页信息来对针对更新发生的 S 块 15 的特定部分的缓冲器 1004 中的 S 日志进行更新，并且新接收的来自例如压缩器的 L 页可以被读到缓冲器 1004 中。当新 L 页在缓冲器中时，逻辑到物理地址转换映射 1002 中的 L 页 100 的条目可以暂时反映其在缓冲器中的位置，如箭头标注“1”所示。随后，在由仍在缓冲器 1004 中的现在被更新的 S 日志中的新创建的条目指定的 E 页地址处，所更新的 L 页 100 可以被刷新到热 S 块 1008。对逻辑到物理地址转换映射 1002 中的映射表条目进行更新以反映物理 E 页地址以及 L 页最终目的地址的偏移，如箭头“2”所建议的。

[0057] 如图 13 所示，在稍后的时间点，例如，在积累了充足数量的新条目之后，易失性存储器缓冲器 1004 中的 S 日志可以被写出到非易失性存储器，例如，到系统 S 块 1010。系统 S 块 1010 可以是由控制器固件分配以用于存储 S 日志的闪速存储器的一部分。在非易失性

存储器中保存 S 日志使逻辑到物理地址转换映射的稍后的重建能够进行，正如所需要的。

[0058] 更新 S 日志和 S 日志映射

[0059] 图 14-17 是示出了根据一个实施例的对系统带中的 S 日志和 S 日志映射进行更新的方法的方面的框图。附图示出了与由对 L 页 100 的相同的示例性更新所触发的 S 日志更新机制相关的额外细节。如其中所示，在易失性存储器（例如，同步动态随机存取存储器(SDRAM)）中的 S 日志映射 1402 可以包含指向 S 日志在非易失性存储器中的物理位置的条目，其中所述 S 日志包含与 L 页 100 相关联的 P2L 映射数据。在该说明性示例中，相关的 S 日志（即，针对存储 L 页 100 的 S 块 15 的特定部分的 S 日志）位于系统 S 块 3(1408 处）的 E 页 42 的起始位置处。

[0060] 其后，如图 15 所示，L 页 100 被更新，其长度为 1534 字节。然后可以从逻辑到物理地址转换映射 1002 中提取与当前 L 页 100 相关的信息（例如，E 页地址、偏移和长度）。其后，存储 L 页 100 的 E 页地址可以用于提取 S 日志号，所述 S 日志号用于定位在 S 日志映射 1402 内的 S 日志条目位置。例如，在其中如图 5 和图 6 处所示的情况下，S 日志号可以包括第一 32 位 E 页地址的 27 个 MSb。所述 S 日志号然后可以用于索引到 S 日志映射 1402 以获得位置，该位置然后可以用于识别包含感兴趣的 S 日志的系统 S 块。其后，如图 15 所示，系统 S 块信息 908 可以被更新以反映以下事实 :S 块 3 处的 S 日志中的针对 L 页 100 的日志条目现在是失效的，这是因为考虑到对 L 页 100 的最近更新，之前由该条目占用的空间现已被解除分配。所解除分配的空间现在是空闲空间，并且必须被记录。可选地，所使用的空间可以被记录，而空闲空间从使用的空间得出。因此，针对 S 块 3 的系统 S 块信息 908 中的条目在空间上增加了一个 S 日志条目 ;即，这里在该示例性实现中开发了 7 字节。该空的空间其后可以在系统带的 GC 期间被考虑。

[0061] 如图 16 所示，L 页 100 然后可以被写到缓冲器 1004，如 1005 处所示，因此，逻辑到物理地址转换映射 1002 可以被更新，以用于指向缓冲器 1004（指示 L 页 100 被存储的物理位置）并且用于存储更新的 L 页 100 的长度（在这里所开发的示例中为 1534)。应当注意的是，在此处，系统 S 块 3 仍包括指向包含 L 页 100 的附图标记 1006 处的 S 块 15 的 S 日志条目。然而，S 块 3 的系统 S 块信息已经被更新以指示 :附图标记 1408 处的系统 S 块 3 内的现已过时的 S 日志条目目前所占用的空间实际上是空闲空间。

[0062] 图 16 也示出了缓冲器中不同的 S 日志。在一个实施例中，缓冲器 1004 中的 S 日志是用于记录写到开放的热 S 块 7 的新 L 页。在该示例中，利用更新的 L 页 100（现在也在缓冲器中）的 L 页号以及其长度信息来更新缓冲器 1004 中的 S 日志（其还没有被刷新到非易失性存储器），如图 16 的 1007 处的箭头所指示的。L 页 100 现在可以从缓冲器 1004 被刷新到当前开放的热 S 块 7，如图 16 的 1404 处所引用的。在此处，缓冲器中的 S 日志被更新以反映 :更新的 L 页 100 已经被写到非易失性存储器。可替换地，可以在更新的 L 页 100 被写到非易失性存储器之前，利用更新的 L 页 100 的地址来更新逻辑到物理地址转换映射 1002 和 S 日志。在一个实施例中，该 S 日志中更新的条目将包含由逻辑到物理地址转换映射 1002 中的新 L 页 100 的条目所指向的 E 页和关于 L 页 100 的额外信息，如图 6 先前所示。

[0063] 如图 17 所示，在积累了充足的数据之后，缓冲器 1004 中的 S 日志可以被写出到开放的系统 S 块 1（在 1406 处所引用的）。在该示例中，该 S 日志被写出到开放的系统 S 块 1 的 E 页 19。S 日志映射 1402 现在包括指示 S 块 15 的部分存储在系统 S 块 3，E 页 42 的一

个条目。然而，指向在 S 块 15 内 L 页 100 的旧位置的在该 S 日志内的条目不再是有效的。S 日志映射 1402 还包括指示 S 块 7 的 S 日志存储在系统 S 块 1，E 页 19 的一个条目。该 S 日志包括指向 1404 处的开放的热 S 块 7 作为更新的 L 页 100 在非易失性存储器中的位置的有效条目。以这种方式，包含 P2L 信息的 S 日志在缓冲器 1004 中被更新，并且最后被写出到非易失性存储器。另外，可以适当地更新 S 日志映射 1402 以使其指向这样的新更新的 S 日志。此外，可以适当地更新逻辑到物理地址转换映射 1002，以使其指向正确的开放的热 S 块。

[0064] 有利地，在新写入时，本文描述和示出的 S 日志结构使写开销最小化。根据一个实施例，新的写操作迫使将新的 S 日志条目写到非易失性存储器中。因为只生成了每 L 页很少量的 S 日志条目数据（例如，在本文中所描述的 7 字节），所以由于系统数据写所导致的 WA 与常规系统相比有所减少。S 日志数据是有效地大量的命令历史记录。更新的 S 日志数据被汇集起来并且顺序地写出到系统带。本文所描述和示出的 S 日志系统是高效的，这是由于所产生的映射信息与要被写入的数据是相同的，而没有额外的未改变的条目，只有被改变的条目被写到非易失性（例如，闪速）存储器，这与逻辑到物理映射的全部或部分被写到非易失性（例如，闪速）存储器不同。

[0065] 上电时，系统带中的所有 S 日志数据可以被读取并且处理到 DRAM 中，以在易失性存储器中重建逻辑到物理地址转换映射。根据一个实施例，这按照 S 块被分配到系统带的顺序被完成。在一个实施例中，硬件支持用于使该大数据的装载能够快速发生以满足上电的时间要求（这在没有硬件的情况下可能是不可行的，因为这本质上是根据命令历史记录来生成逻辑到物理地址转换映射）。根据一个实施例，S 日志映射还在上电时被构建以使其指向系统带中存储的有效的 S 日志。

[0066] 用户数据 S 块的垃圾收集

[0067] 图 18-21 是示出了根据一个实施例的垃圾收集的方面的框图。如其中所示，可以扫描用户 S 块信息 906 中的数据以选择“最好的”S 块用于垃圾收集。存在若干标准，可以对所述若干标准进行评估以选择哪个 S 块用于垃圾收集。例如，用于垃圾收集的最好的 S 块可以是具有最大量的空闲空间和最低程序擦除 (PE) 量的 S 块。可替换地，可以对这些和 / 或其他标准进行加权以选择要进行垃圾收集的 S 块。为了示例的目的，在图 18-21 中被选择用于进行垃圾收集的 S 块是 S 块 15，S 块 15 由用户 S 块信息 906 内的信息条目 15 所引用，示出了一些量 +3012 字节的空闲空间（额外的 3012 字节用于记录最近废弃的 L 页 100）。应当注意的是，用户 S 块信息 906 可以包括每一个所跟踪的 S 块所经历的 PE 周期数量的运行计数，可以对所述运行计数进行评估以决定哪个 S 块用于垃圾收集，连同其他项的信息一起。如 1006 处所示，S 块 15 具有有效数据（混编的块）和无效数据（非混编的块）的混合。

[0068] 既然已经选择 S 块 15 用于 GC，那么可以查阅（例如，由 S 块号索引到）S 日志映射（参看图 9B 中 912）以找到该 S 块的对应的 S 日志在非易失性存储器中的位置（例如，E 页地址）。为了示出示例，使用 S 日志号 (E 页地址的 27 个 MSb) 来定位由 S 日志映射 912 所指向的一个 S 日志的位置，并且将所述 S 日志读入到缓冲器 1004，如图 18 所示。即，访问由 S 日志映射 912 所指向的系统 S 块 1804 中的 1 个或更多个 E 页，并且可以将在该位置开始存储的 S 日志读入到缓冲器 1004。在一个实施例中，S 日志映射页还包含 S 日志的长

度，这是因为S日志可以跨越一个或多个E页。S日志可以非常大，并且S日志因此可以按片来读取并被处理为可用。

[0069] 其后，缓冲器1004中的S日志中的每一个P2L条目然后可以与逻辑到物理地址转换映射1802中对应的条目进行比较。对于缓冲器1004中的S日志中的每一个条目，可以确定该条目的L页的物理地址是否与逻辑到物理地址转换映射1802中对应的条目中的相同L页的物理地址相匹配。如果二者匹配，那么S日志中的该条目有效。相反地，如果S日志中的L页的地址与逻辑到物理地址转换映射中该L页的条目不匹配，那么S日志中的该条目无效。

[0070] 根据一个实施例，由于在对其条目进行分析和比较的S日志中找到了有效条目，所以其所引用的L页可以从S块15中被读出并且写到缓冲器1004，如图19所示。可以使用相同的处理以用于覆盖S块15的其他S日志直到整个S块被处理为止。同样如图19中附图标记1006处所示，S块15现在仅包含无效数据。这是因为由针对S块15的S日志中的条目指示为有效的数据已经被保留，并且不久将被移动到新的S块。在所示示例中，由于系统S块1804中的S块15的示例S日志中的条目指向这样的无效数据，所以该S日志被示出为被混编以指示其现在是陈旧的。然后可以更新逻辑到物理地址转换映射1802，生成被读取到缓冲器1004的有效数据的新的E页起始地址。应当注意的是，在对逻辑到物理地址转换映射的更新期间，可以针对有效条目来重新核对该映射并且可以在映射更新过程期间锁定该映射以保证原子性。有效数据还将迫使新的S日志条目在S日志1005中生成，这在一个实施例中是针对冷S块1801的。

[0071] 在一个实施例中，然后可以将有效数据写出到冷S块1801（热S块用于最近写入的主机数据，不用于垃圾收集数据），如图20所示。在某个稍后的时刻（例如，在增加了充足数量的条目之后），S日志1005可以被写出到系统带中的系统S块1804。S块15现已进行了垃圾收集，并且用户S块信息906现在指示整个S块15是空闲空间。其后S块15可以被擦除，其PE计数被更新并且成为对新的写入可用。应当注意的是，无效的S日志仍存在于系统S块1804中。在某个稍后的时刻可以对由该无效S日志所占用的系统带中的闪速存储器中的空间进行垃圾收集。

[0072] 系统S块上的垃圾收集

[0073] 图22-26是示出了根据一个实施例的对系统块进行垃圾收集的方面的框图。在图22-26发展的示例中，假设图22中的附图标记2208处所示的系统S块3已经被挑选用于垃圾收集。一旦系统S块被挑选，则在所挑选的系统S块中的所有E页（并且通过扩展，包含在其中的所有S日志）可以被读取（顺序地或非顺序地）到缓冲器1004。

[0074] 如图23所建议的，被读取到缓冲器1004的一个或多个S日志的S日志号然后可以从该S日志的头部提取。然后每一个这样的系统S日志号可以用于在S日志映射2202中进行查找，以确定对应的S日志是否仍然有效。根据一个实施例，无效的S日志是其S日志号不能与S日志映射2202中的对应的条目相匹配的那些S日志，其中S日志映射2202中对应的条目具有关于S日志物理地存储于何处的最近更新的信息。例如，如果在S日志映射2202中S日志号“12345”的条目指向系统S块120内的E页，那么S块3（该S块正在进行垃圾收集）中的S日志“12345”的拷贝是废弃的。同样地，如果相反S日志映射条目指向来自S日志“12345”当前所驻留的S块3的E页，那么S日志“12345”仍然有效。

[0075] 在一个实施例中，进行垃圾收集的有效 S 日志可以包括有效和无效条目的混合，因此需要对条目进行单独检查。如图 24 所示，然后可以将每一个有效的 S 日志 2402 中的每一个条目与存储器中的逻辑到物理地址转换映射 1802 中的对应条目进行匹配。即，可以将每一个 S 日志条目所引用的 L 页的 E 页地址与逻辑到物理地址转换映射 1802 中指定的 L 页的 E 页地址进行比较。如果二者匹配，那么该 S 日志条目是有效的。根据一个实施例，将 S 日志中的 E 页地址和在 E 页内的偏移与逻辑到物理地址转换映射 1802 中指定的 L 页的 E 页地址和在 E 页内的偏移进行比较，是必要的。相反地，如果 S 日志中的 L 页的 E 页地址（或 E 页地址和偏移）与地址转换映射 1802 中的该 L 页的条目中的 E 页地址（或 E 页地址和偏移）不匹配，那么所述 S 日志中的该条目是无效的。

[0076] 根据一个实施例，由于有效条目在 S 日志 2402（对其条目进行分析和比较）中被识别，则可以将有效条目拷贝到 S 日志 2402 的新版本 2502，如图 25 所示。根据一个实施例，S 日志 2502 可以具有与 S 日志 2402 相同的 S 日志号。以这种方式，从被挑选用于 GC 的 S 块中加载到缓冲器 1004 中的每一个 S 日志可以首先在日志级被确定为有效或无效，并且然后，如果被确定为有效，则被压缩到只包含有效条目。根据一个实施例，无效的 S 日志条目简单地不被拷贝到 S 日志 2502 的新版本。因此，假定 S 日志 2402 具有一个或多个废弃的条目，可以预期的是，S 日志的新版本 2502 将小于其旧版本 2402（包括更少的条目）。应当注意的是，如果 S 日志具有所有有效的条目，则其新版本的尺寸将保持相同。

[0077] 如图 26 所示，然后可以将 S 日志 2502（现在是无效或废弃的 S 日志 2402 的新版本）写到当前开放的系统 S 块，在图 26 中其为开放系统 S 块 1，附图标记 2206。其后，可以利用 S 日志 2502 在开放的系统 S 块 1 中的新位置来对 S 日志映射 2202 进行更新。在该过程的结尾，系统 S 块 3(2208) 中的 S 日志已经进行了垃圾收集，并且系统 S 块 3 然后可以被擦除并成为对于未来的编程可用。

[0078] 尽管已经描述了本公开的特定实施例，但这些实施例只是以示例的方式进行呈现，而并不是要限制本公开的范围。实际上，可以以多种其他形式来体现本文所描述的新颖的方法、设备和系统。另外，可以对本文所描述的方法和系统的形式做出各种省略、替换和改变，而不偏离本公开的精神。所附的权利要求及其等价物是要涵盖这样的形式或修改，其将落入本公开的范围和精神。例如，本领域中的技术人员将意识到，在各种实施例中，实际的物理和逻辑结构可以与附图中所示的物理和逻辑结构不同。取决于实施例，可以移除在以上示例中所描述的特定步骤，可以添加其他步骤。同样，上文所公开的特定实施例的特征和属性可以以不同的方式组合以形成另外的实施例，其全部落入本公开的范围内。尽管本公开提供了特定优选的实施例和应用，但对本领域的普通技术人员而言显而易见的是，包括没有提供本文所阐述的全部特征和优点的实施例在内的其他实施例，也在本公开的范围内。因此，本公开的范围是要仅通过参考所附的权利要求来进行限定。

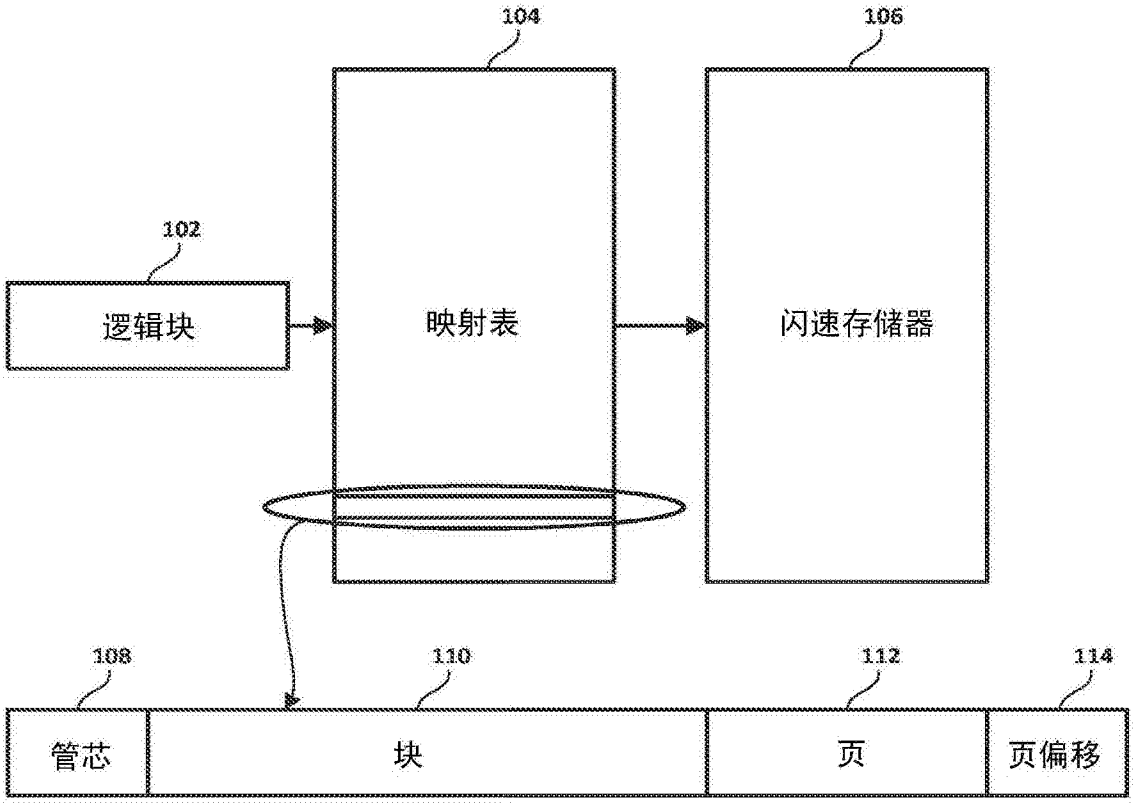


图 1（现有技术）

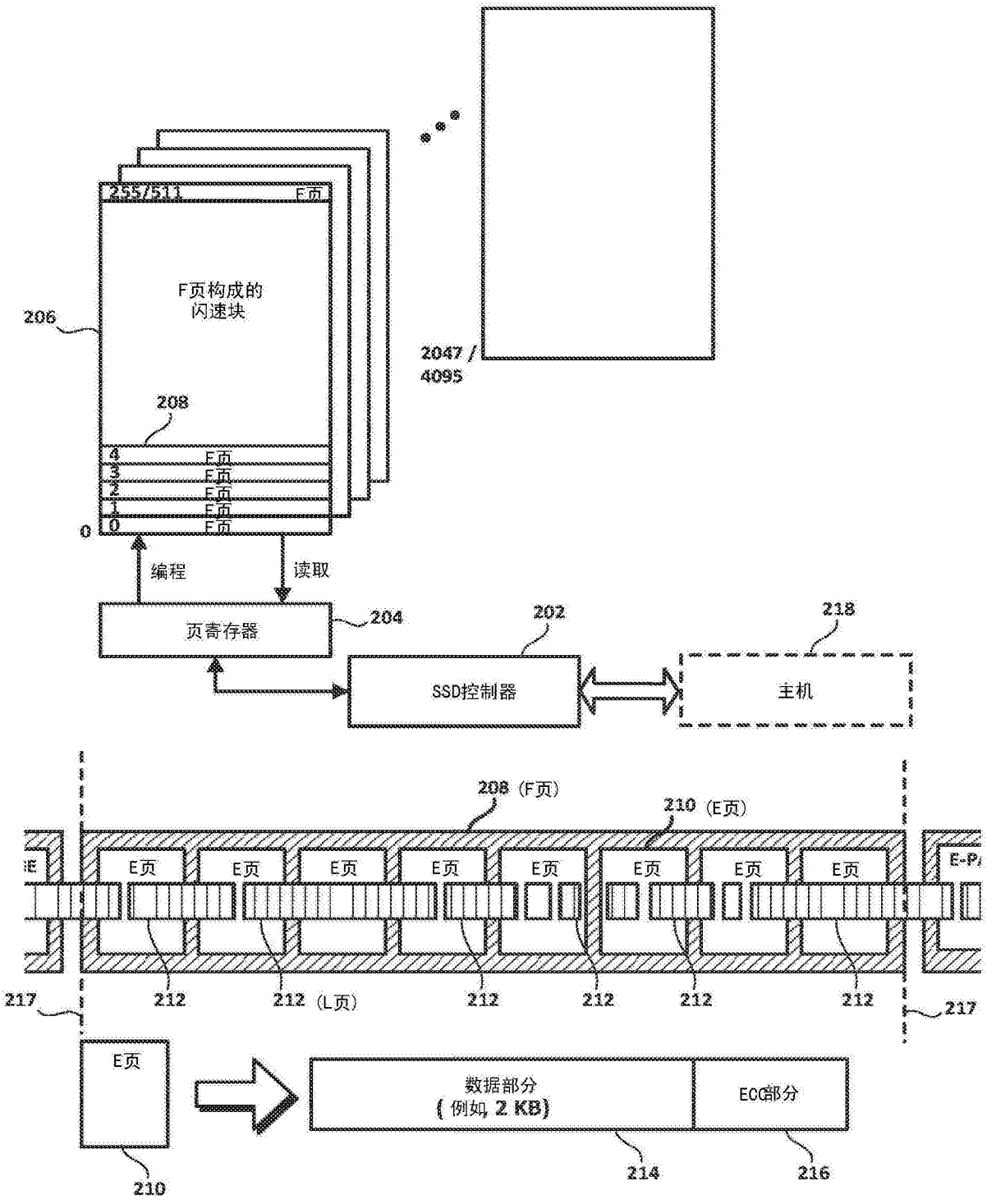
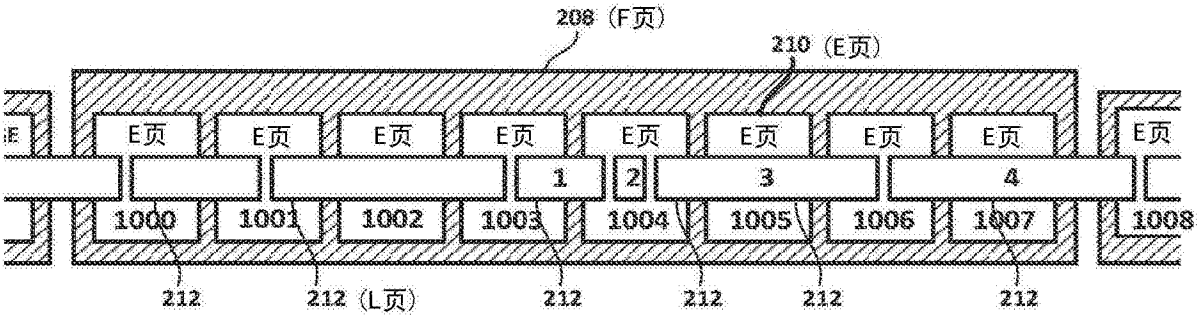

255/511
F页
F页构成的
闪速块
206
208
4
3
2
1
0
F页
编程
读取
页寄存器
204
202
SSD控制器
218
主机
2047 /
4095
208 (F页)
210 (E页)
E页
212
212 (L页)
217
E页
数据部分
(例如, 2 KB)
ECC部分
214
216
210

图 2



L页#			需要的E页
L页1=E页 1003	偏移 800	长度 1,624	1003,1004
L页2=E页 1004	偏移 400	长度 696	1004
L页3=E页 1004	偏移 1,120	长度 4,096	1004, 1005, 1006
L页4=E页 1006	偏移 1,144	长度 3,128	1006, 1007, 1008

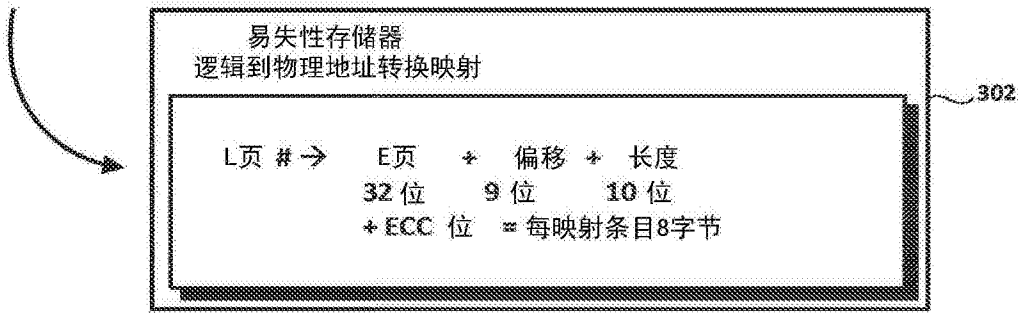


图 3

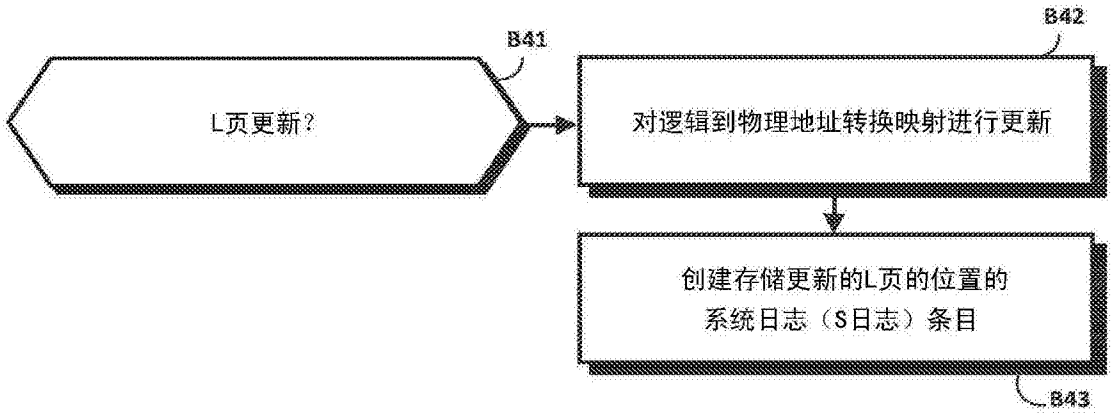


图 4

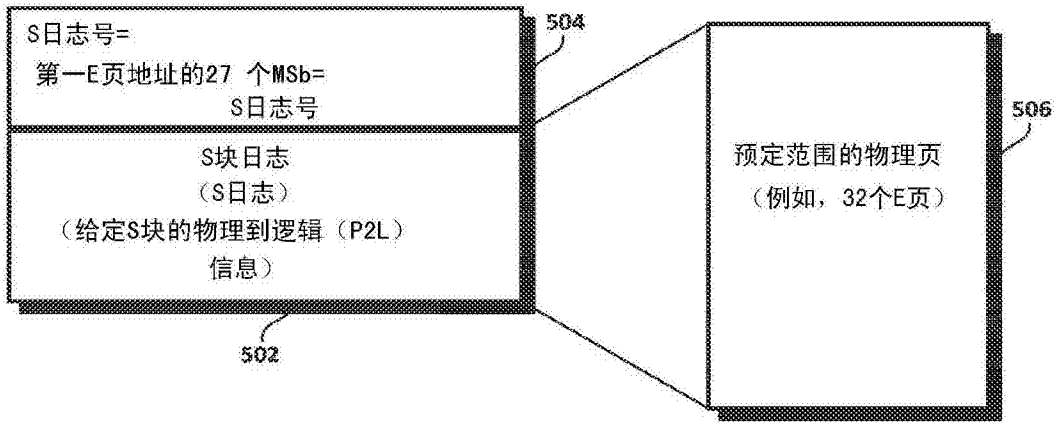

S日志号=
第一E页地址的27 个MSb=
S日志号
S块日志
（S日志）
（给定S块的物理到逻辑（P2L）
信息）
502
504
506
预定范围的物理页
（例如，32个E页）

图 5

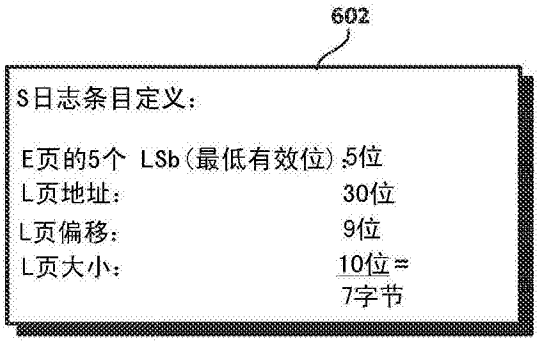

602
S日志条目定义：
E页的5个 LSb（最低有效位）：5位
L页地址： 30位
L页偏移： 9位
L页大小： 10位≈
7字节

图 6

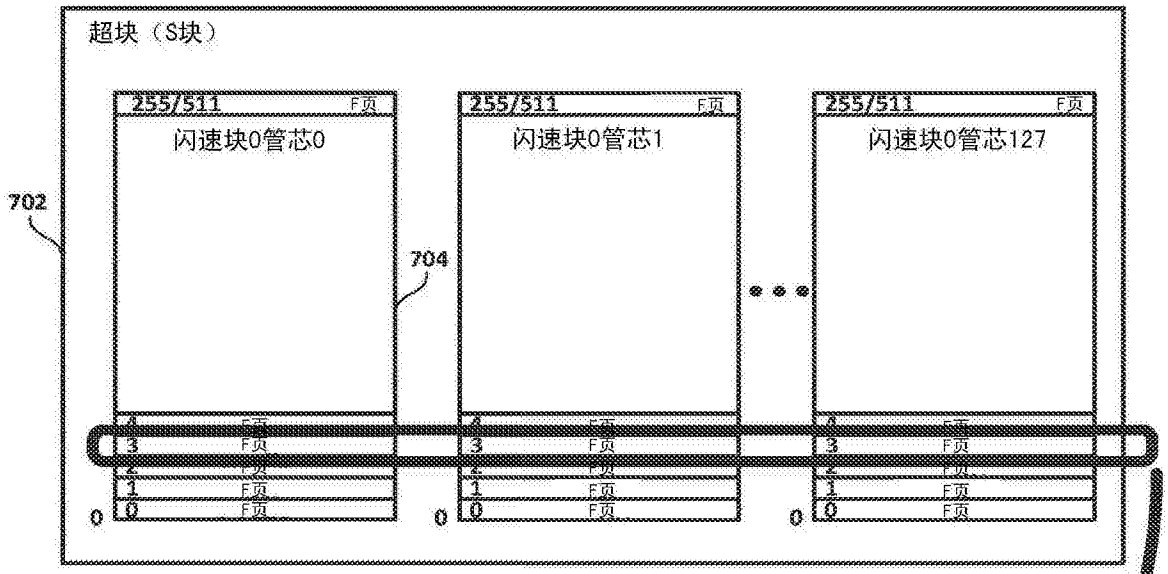

超块（S块）
255/511
F页
闪速块0管芯0
闪速块0管芯1
闪速块0管芯127
702
704
4
3
2
1
0

图7

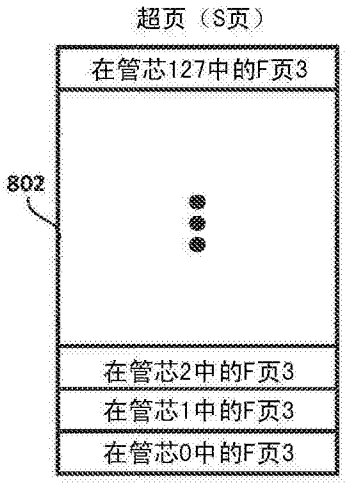

超页（S页）
在管芯127中的F页3
802
在管芯2中的F页3
在管芯1中的F页3
在管芯0中的F页3

图8

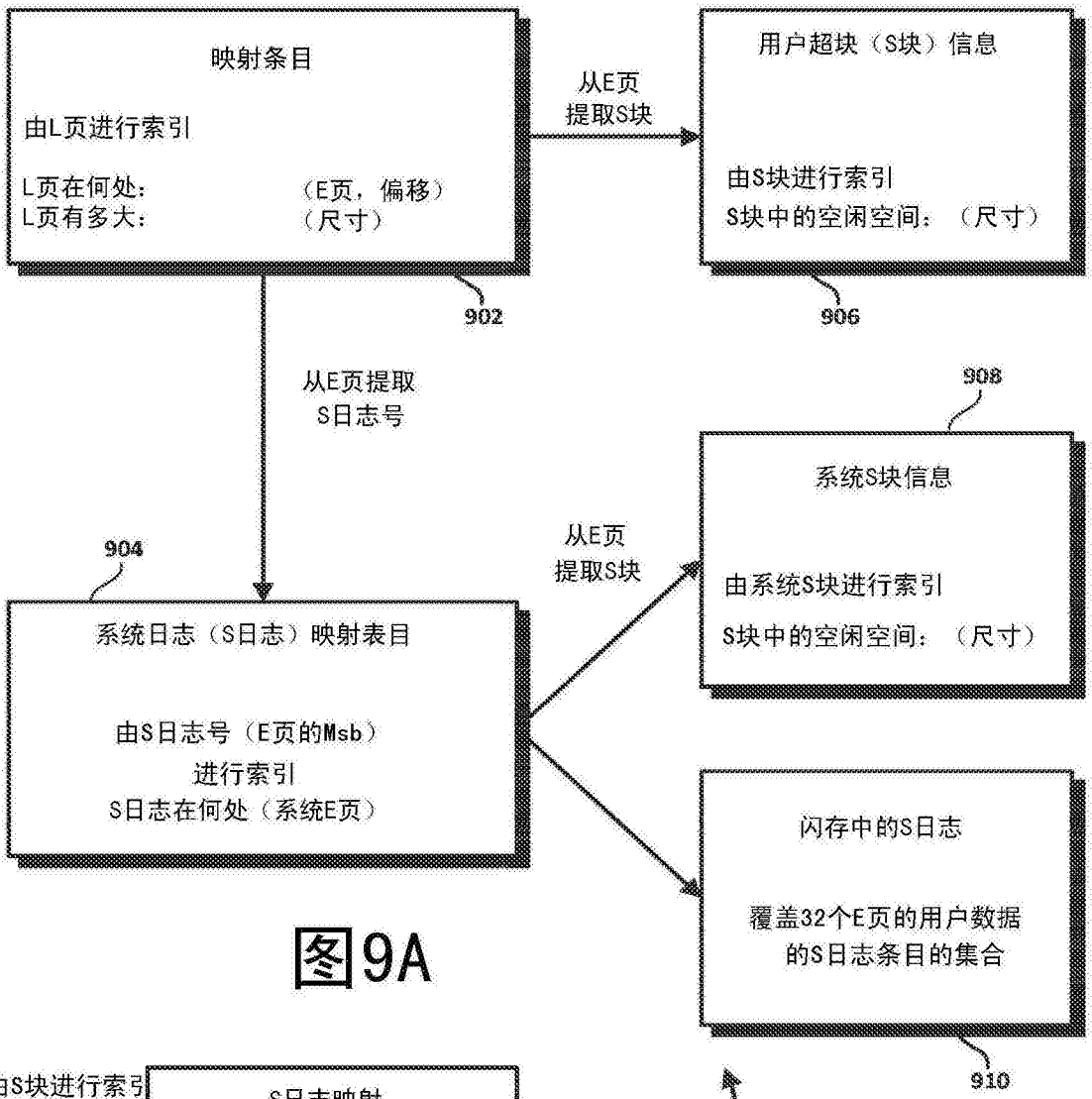

映射条目
由L页进行索引
L页在何处：（E页，偏移）
L页有多大：（尺寸）
902
从E页
提取S块
用户超块（S块）信息
由S块进行索引
S块中的空闲空间：（尺寸）
906
从E页提取
S日志号
908
系统S块信息
由系统S块进行索引
S块中的空闲空间：（尺寸）
904
系统日志（S日志）映射表目
由S日志号（E页的Msb）
进行索引
S日志在何处（系统E页）
从E页
提取S块
闪存中的S日志
覆盖32个E页的用户数据
的S日志条目的集合
910

图9A

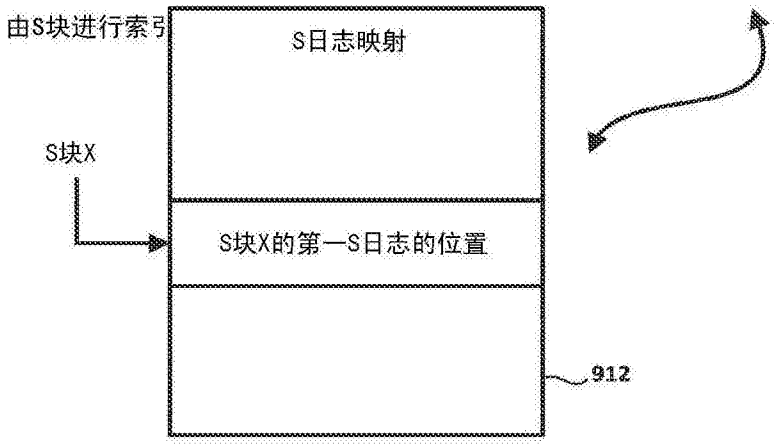

由S块进行索引
S日志映射
S块X
S块X的第一S日志的位置
912

图9B

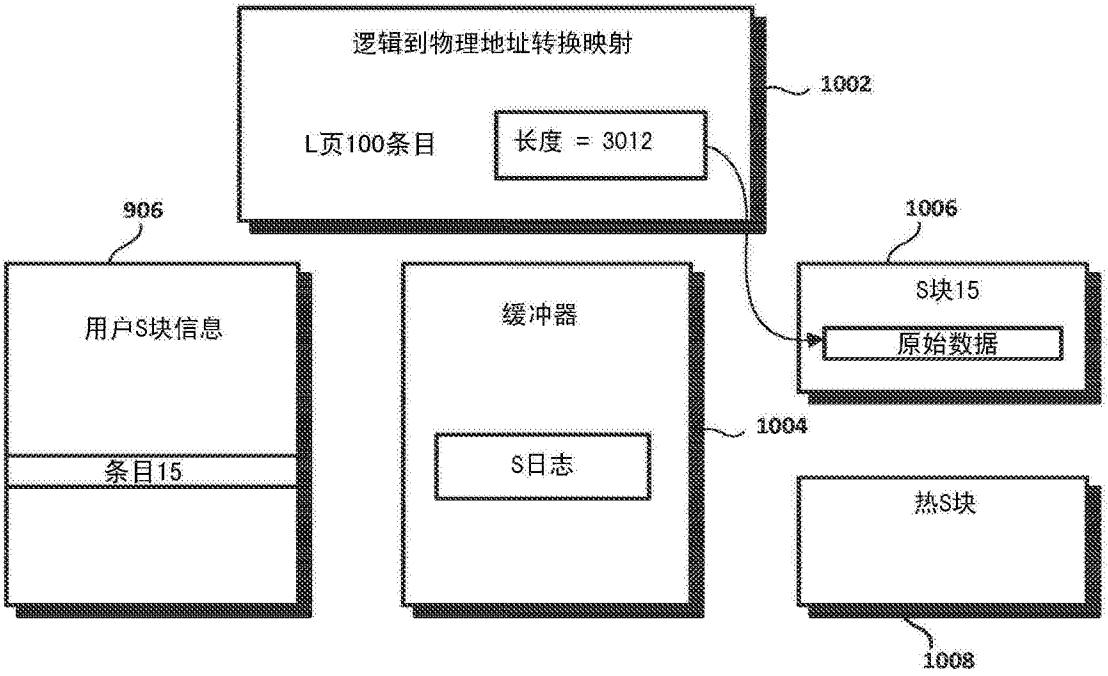

逻辑到物理地址转换映射
1002
L页100条目
长度 = 3012
906
用户S块信息
条目15
缓冲器
S日志
1004
1006
S块15
原始数据
热S块
1008

图 10

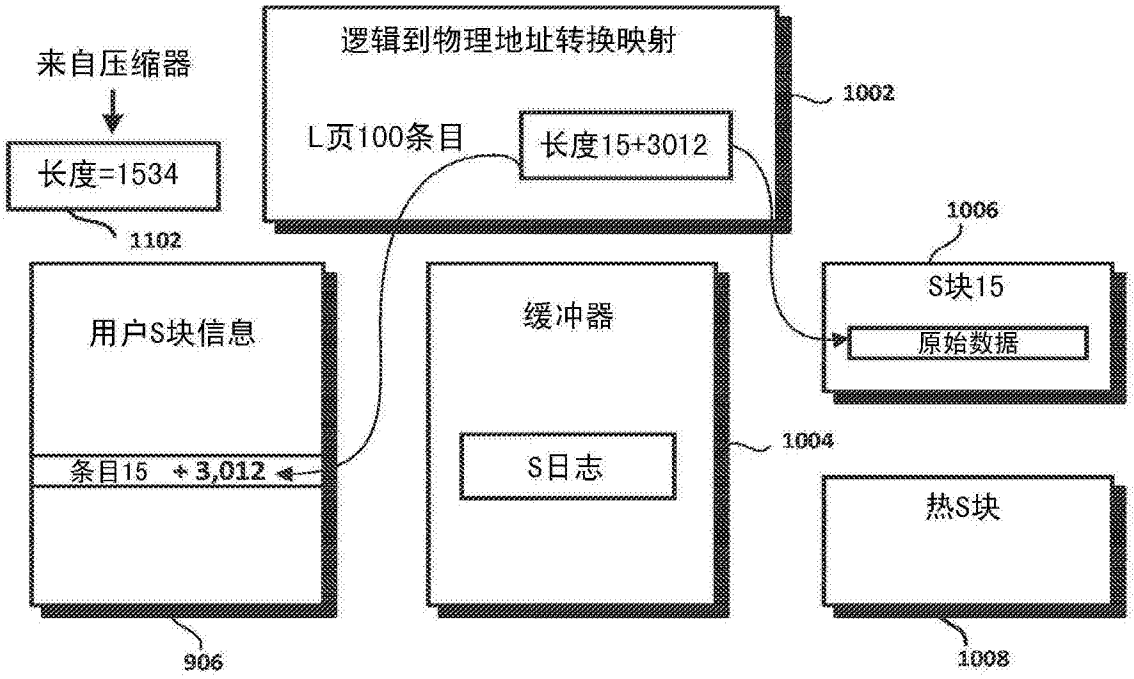

来自压缩器
长度=1534
1102
逻辑到物理地址转换映射
1002
L页100条目
长度15+3012
1006
S块15
原始数据
用户S块信息
条目15 + 3,012
缓冲器
S日志
1004
热S块
906
1008

图 11

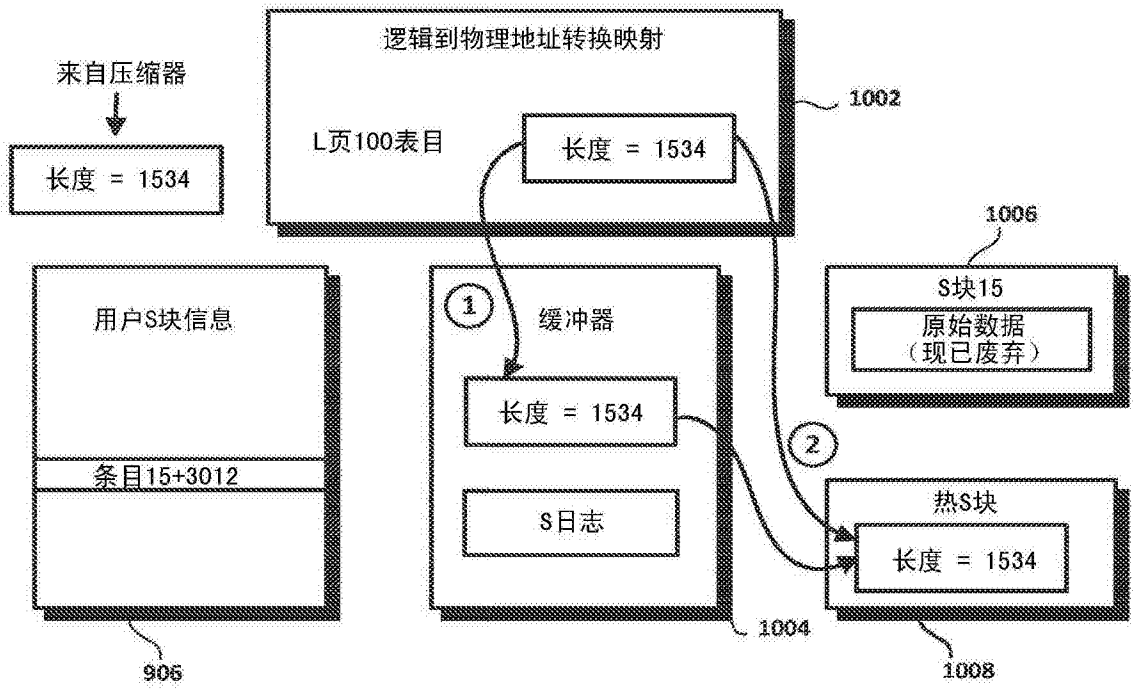

来自压缩器
长度 = 1534
逻辑到物理地址转换映射
L页100表目
长度 = 1534
1002
1006
S块15
原始数据
（现已废弃）
用户S块信息
条目15+3012
906
1
缓冲器
长度 = 1534
S日志
1004
2
热S块
长度 = 1534
1008

图 12

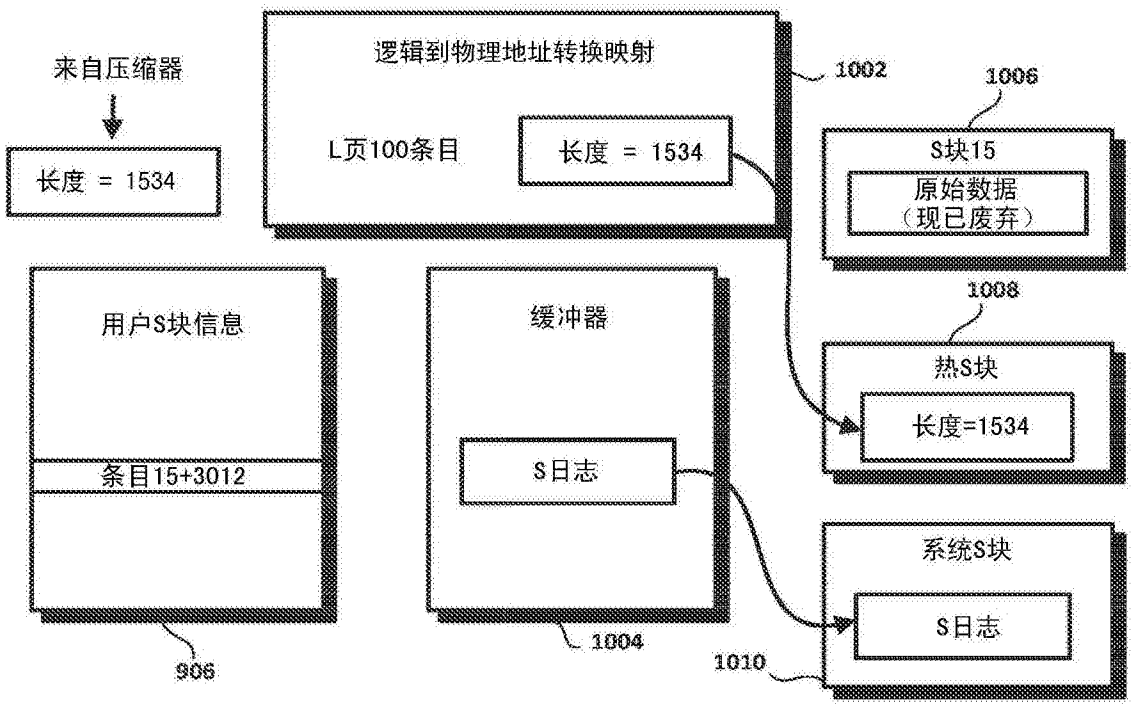

来自压缩器
长度 = 1534
逻辑到物理地址转换映射
L页100条目
长度 = 1534
1002
1006
S块15
原始数据
（现已废弃）
1008
热S块
长度=1534
用户S块信息
条目15+3012
906
缓冲器
S日志
1004
系统S块
S日志
1010

图 13

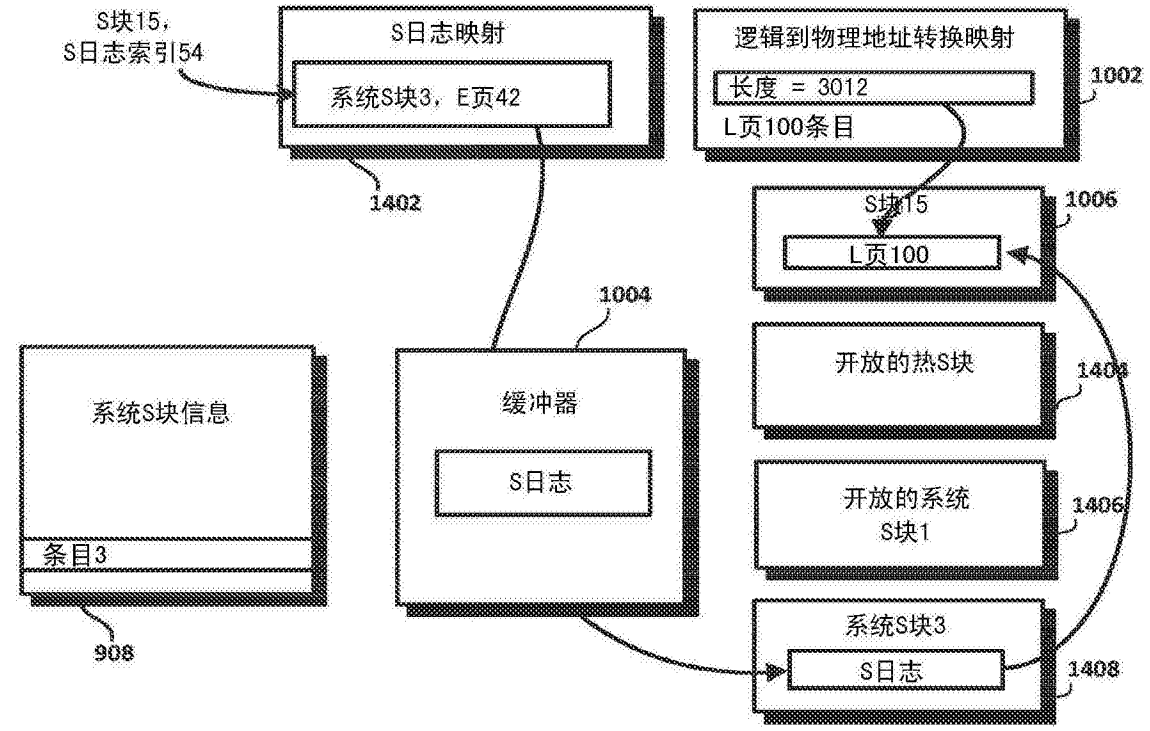

S块15，
S日志索引54
S日志映射
系统S块3，E页42
1402
逻辑到物理地址转换映射
长度 = 3012
L页100条目
1002
S块15
L页100
1006
1004
系统S块信息
条目3
908
缓冲器
S日志
开放的热S块
1404
开放的系统
S块1
1406
系统S块3
S日志
1408

图 14

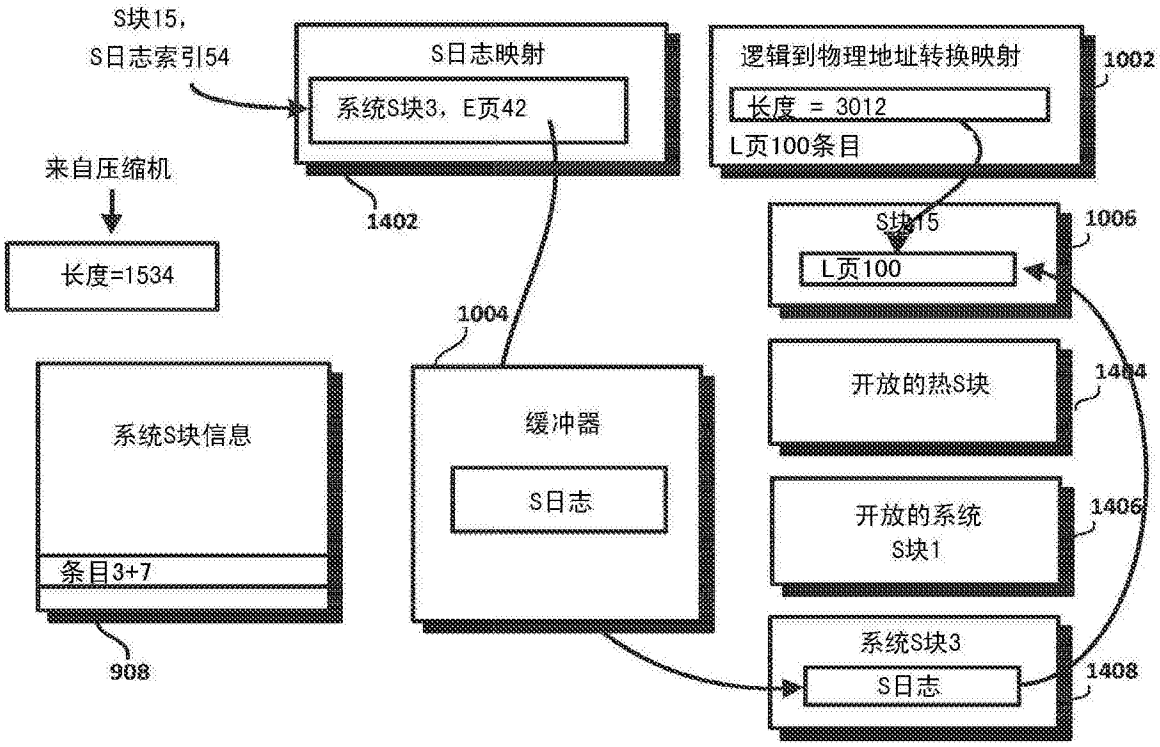

S块15，
S日志索引54
S日志映射
系统S块3，E页42
1402
逻辑到物理地址转换映射
长度 = 3012
L页100条目
1002
来自压缩机
长度=1534
S块15
L页100
1006
1004
系统S块信息
条目3+7
908
缓冲器
S日志
开放的热S块
1404
开放的系统
S块1
1406
系统S块3
S日志
1408

图 15

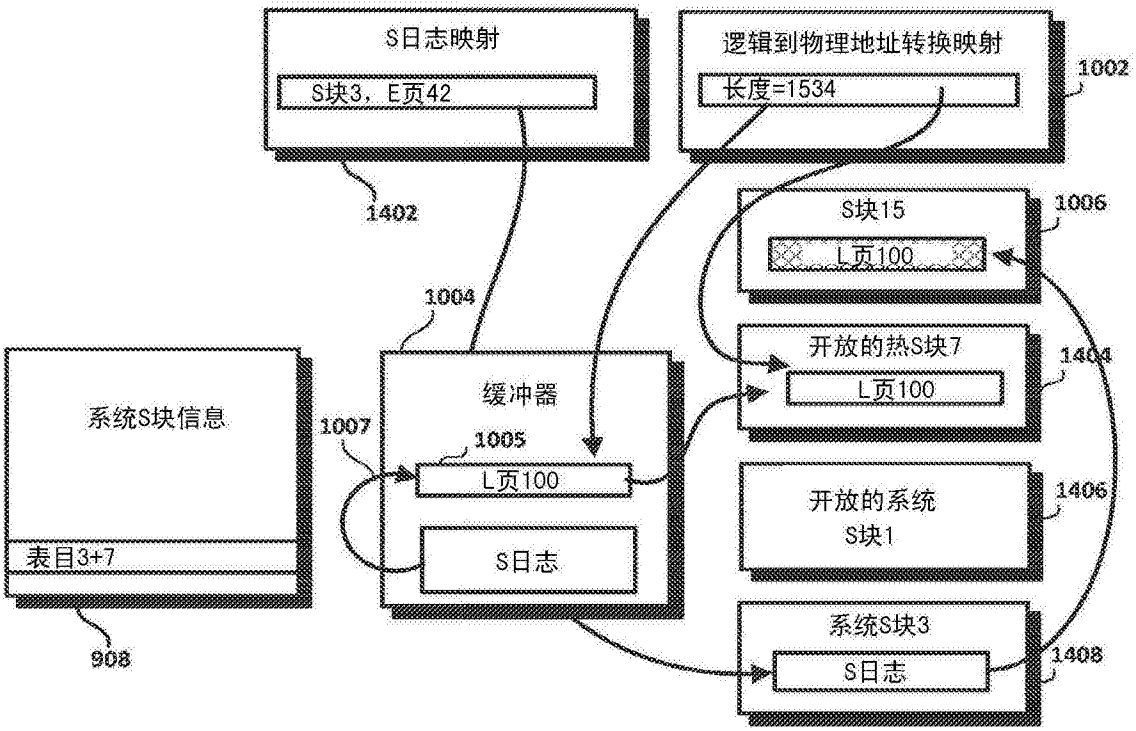

S日志映射
S块3，E页42
1402
逻辑到物理地址转换映射
长度=1534
1002
S块15
L页100
1006
1004
开放的热S块7
L页100
1404
系统S块信息
1007
缓冲器
1005
L页100
S日志
开放的系统
S块1
1406
表目3+7
908
系统S块3
S日志
1408

图 16

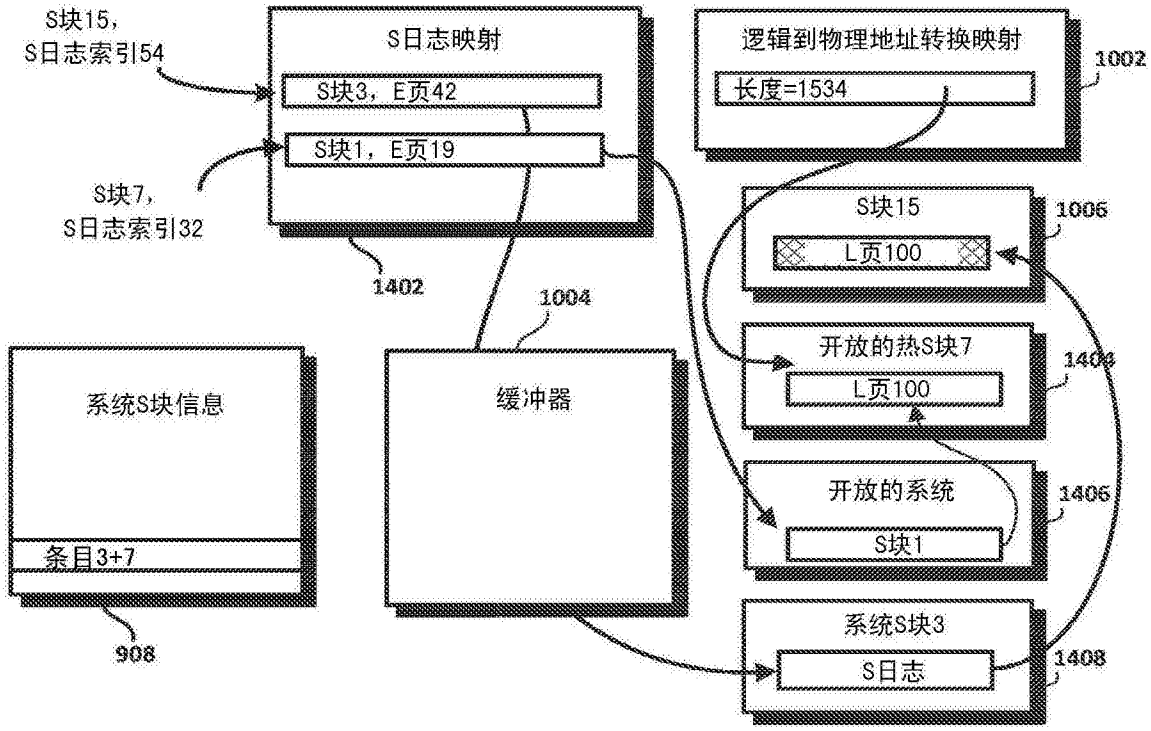

S块15，
S日志索引54
S日志映射
S块3，E页42
S块1，E页19
逻辑到物理地址转换映射
长度=1534
1002
S块7，
S日志索引32
S块15
L页100
1006
1402
1004
开放的热S块7
L页100
1404
系统S块信息
缓冲器
开放的系统
S块1
1406
条目3+7
908
系统S块3
S日志
1408

图 17

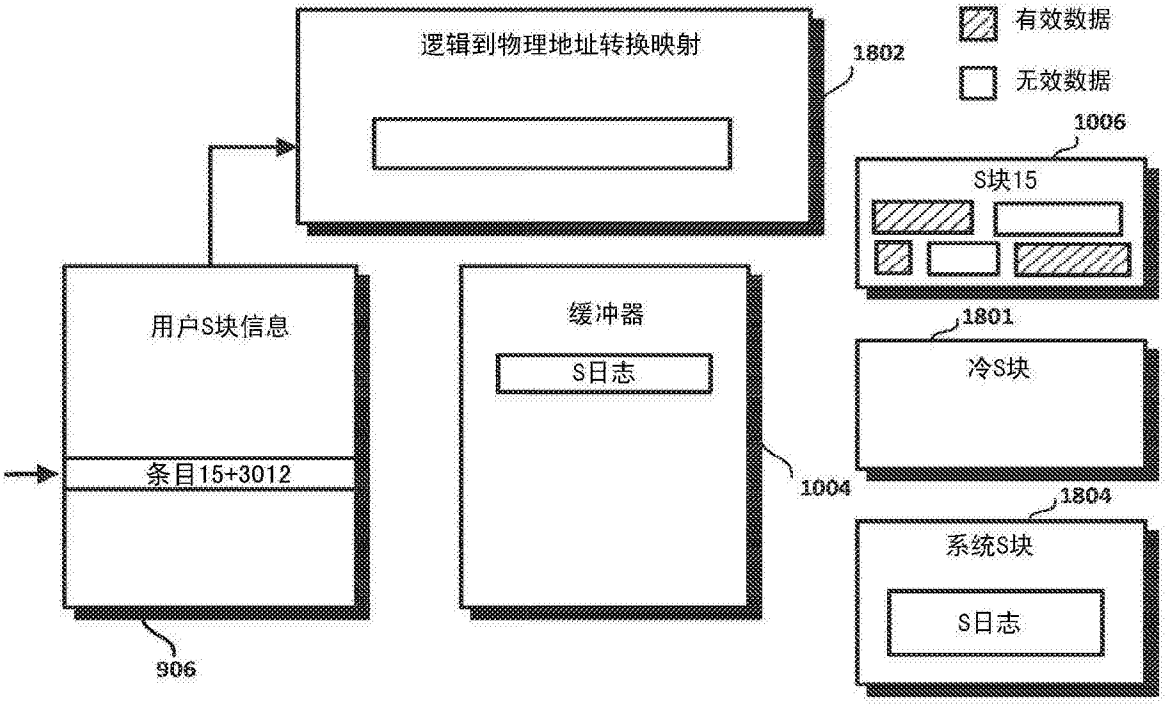

逻辑到物理地址转换映射
1802
有效数据
无效数据
1006
S块15
用户S块信息
条目15+3012
906
缓冲器
S日志
1004
1801
冷S块
1804
系统S块
S日志

图 18

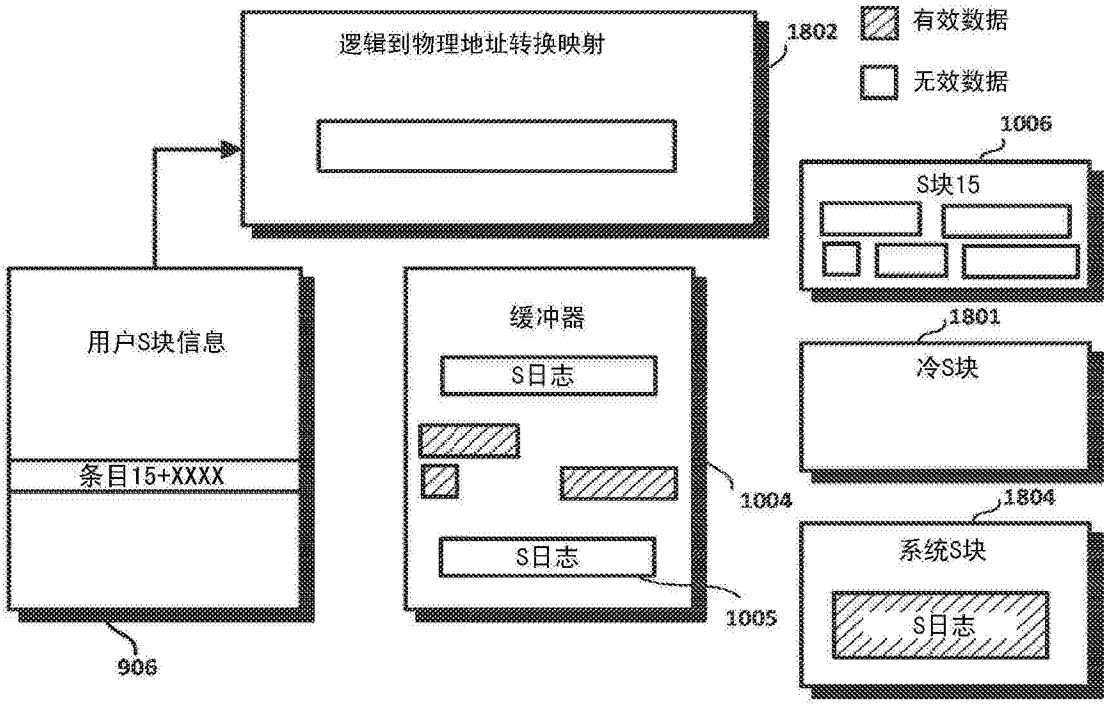

逻辑到物理地址转换映射
1802
有效数据
无效数据
1006
S块15
用户S块信息
条目15+XXXX
906
缓冲器
S日志
S日志
1004
1005
1801
冷S块
1804
系统S块
S日志

图 19

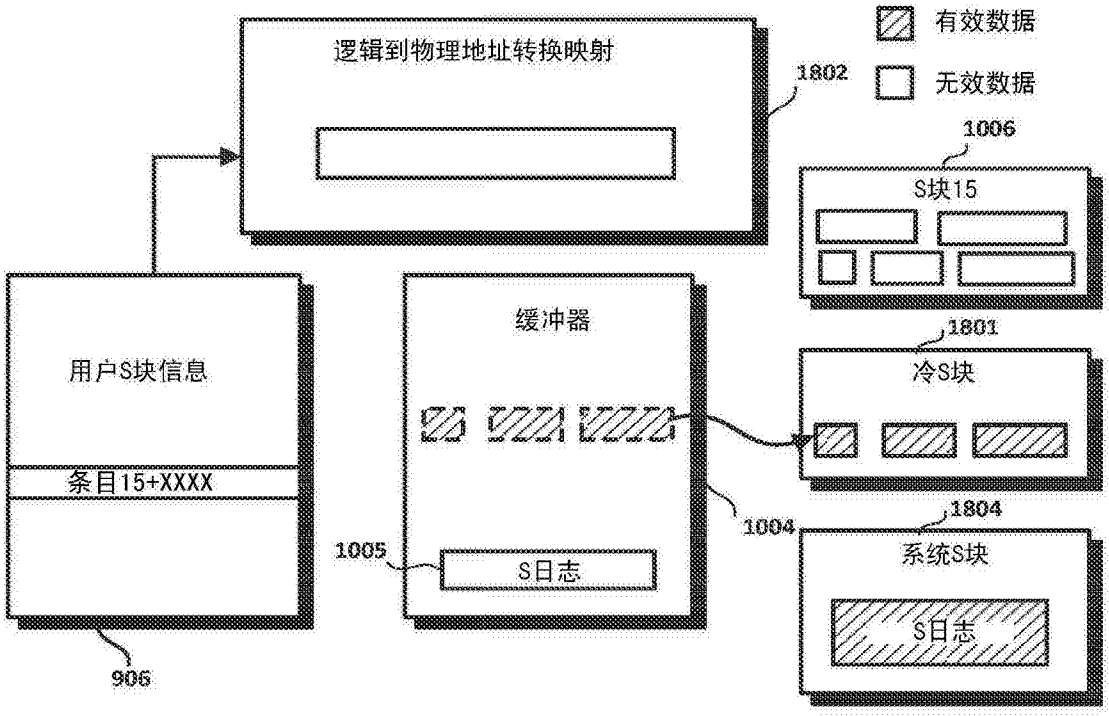

逻辑到物理地址转换映射
1802
有效数据
无效数据
1006
S块15
用户S块信息
条目15+XXXX
906
缓冲器
1005
S日志
1801
冷S块
1004
1804
系统S块
S日志

图 20

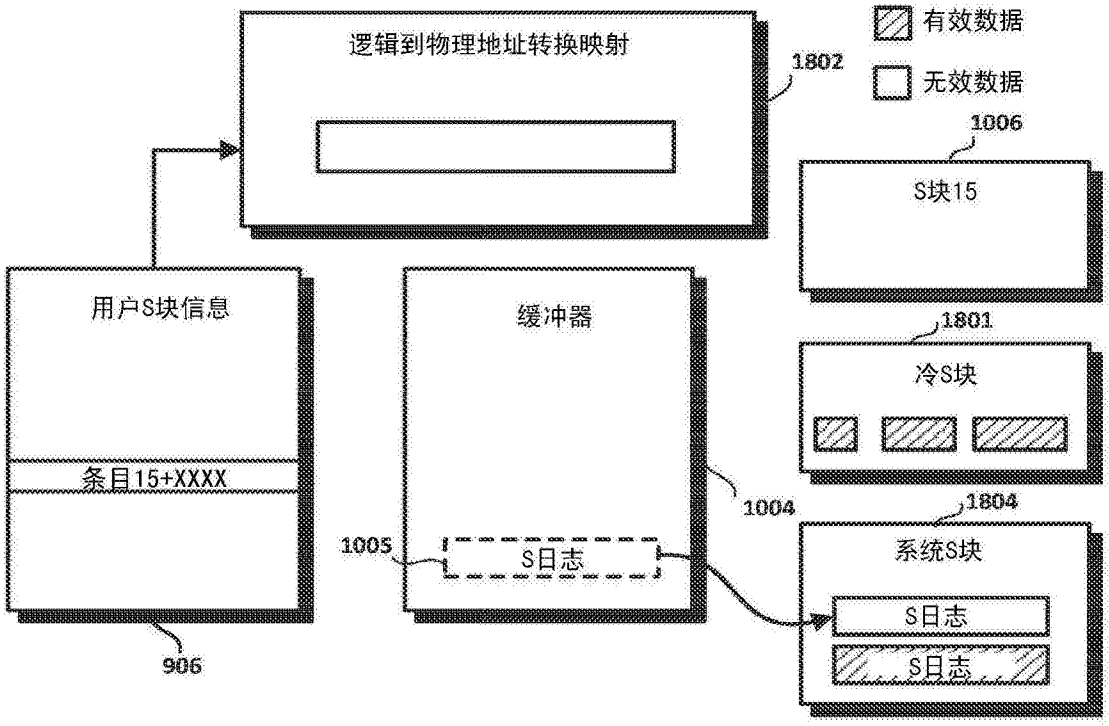

逻辑到物理地址转换映射
1802
有效数据
无效数据
1006
S块15
用户S块信息
条目15+XXXX
906
缓冲器
1005
S日志
1801
冷S块
1004
1804
系统S块
S日志
S日志

图 21

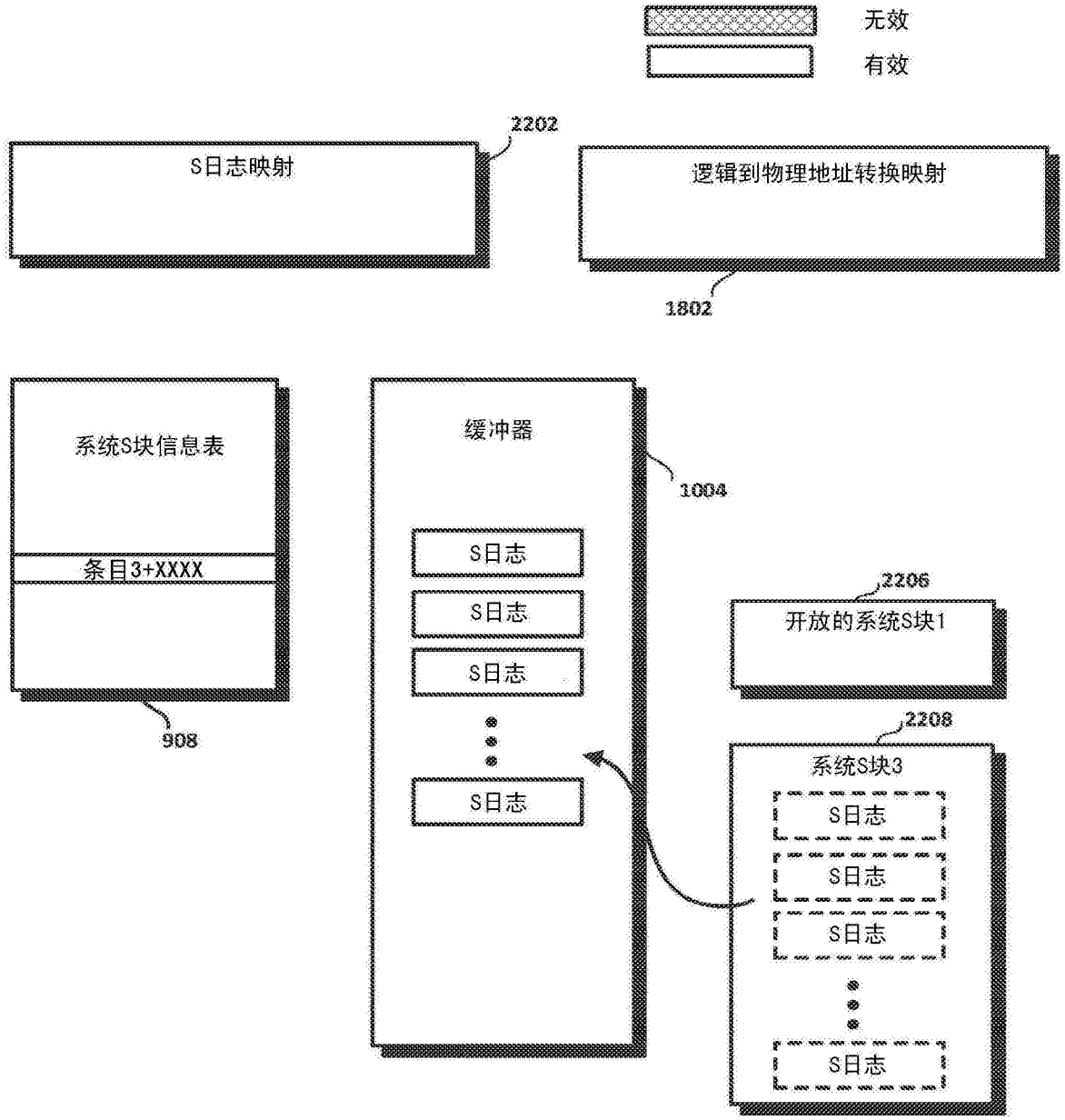

无效
有效
2202
S日志映射
逻辑到物理地址转换映射
1802
系统S块信息表
条目3+XXXX
908
缓冲器
1004
S日志
S日志
S日志
S日志
2206
开放的系统S块1
2208
系统S块3
S日志
S日志
S日志
S日志

图 22

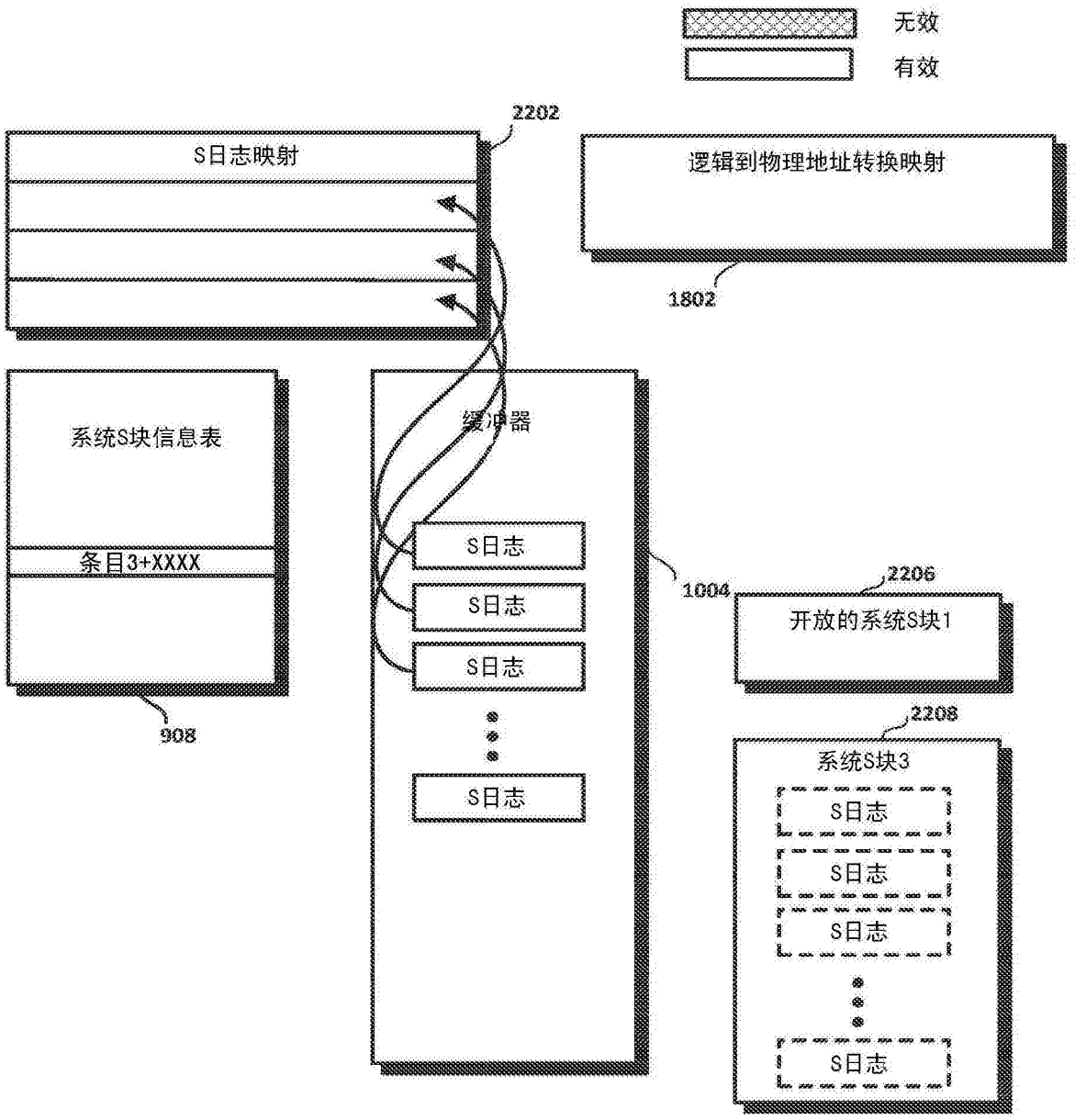

无效
有效
2202
S日志映射
逻辑到物理地址转换映射
1802
系统S块信息表
条目3+XXXX
908
缓冲器
S日志
S日志
S日志
S日志
1004
2206
开放的系统S块1
2208
系统S块3
S日志
S日志
S日志
S日志

图 23

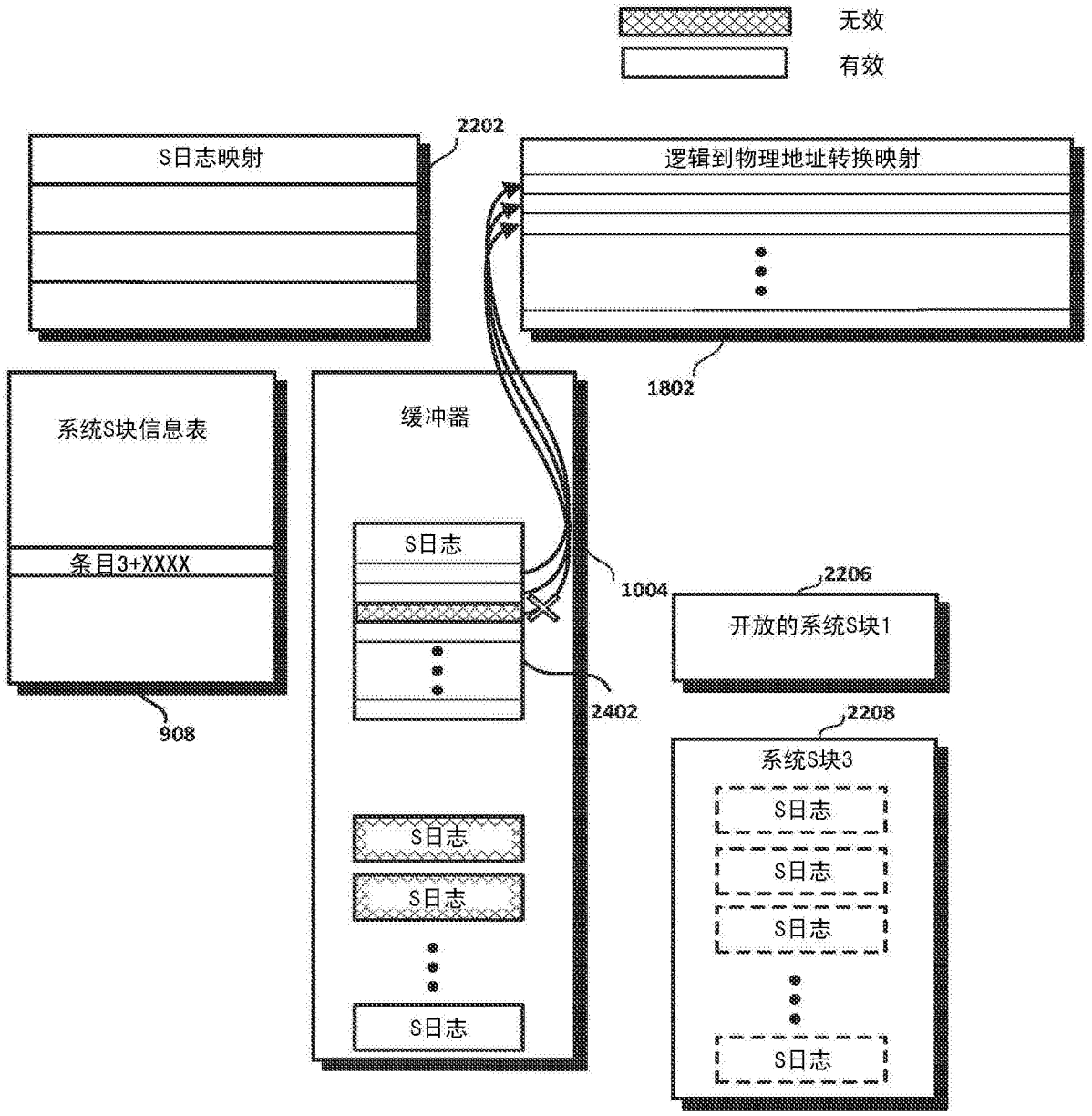

无效
有效
2202
S日志映射
逻辑到物理地址转换映射
1802
系统S块信息表
条目3+XXXX
908
缓冲器
S日志
1004
2206
开放的系统S块1
2402
2208
系统S块3
S日志
S日志
S日志
S日志
S日志
S日志

图 24

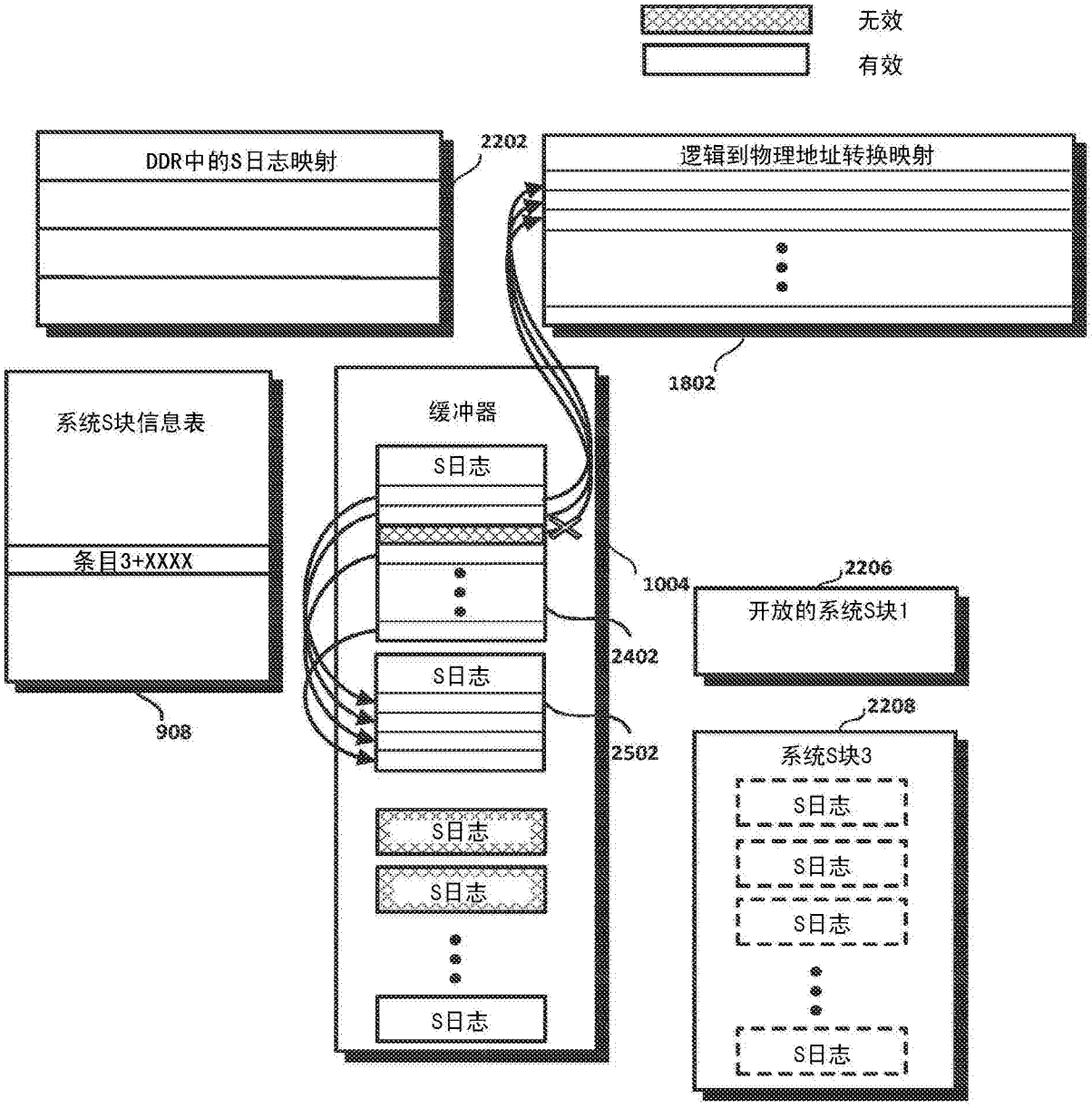

无效
有效
DDR中的S日志映射
2202
逻辑到物理地址转换映射
1802
系统S块信息表
条目3+XXXX
908
缓冲器
S日志
1004
2402
S日志
2502
S日志
S日志
S日志
2206
开放的系统S块1
2208
系统S块3
S日志
S日志
S日志
S日志

图 25

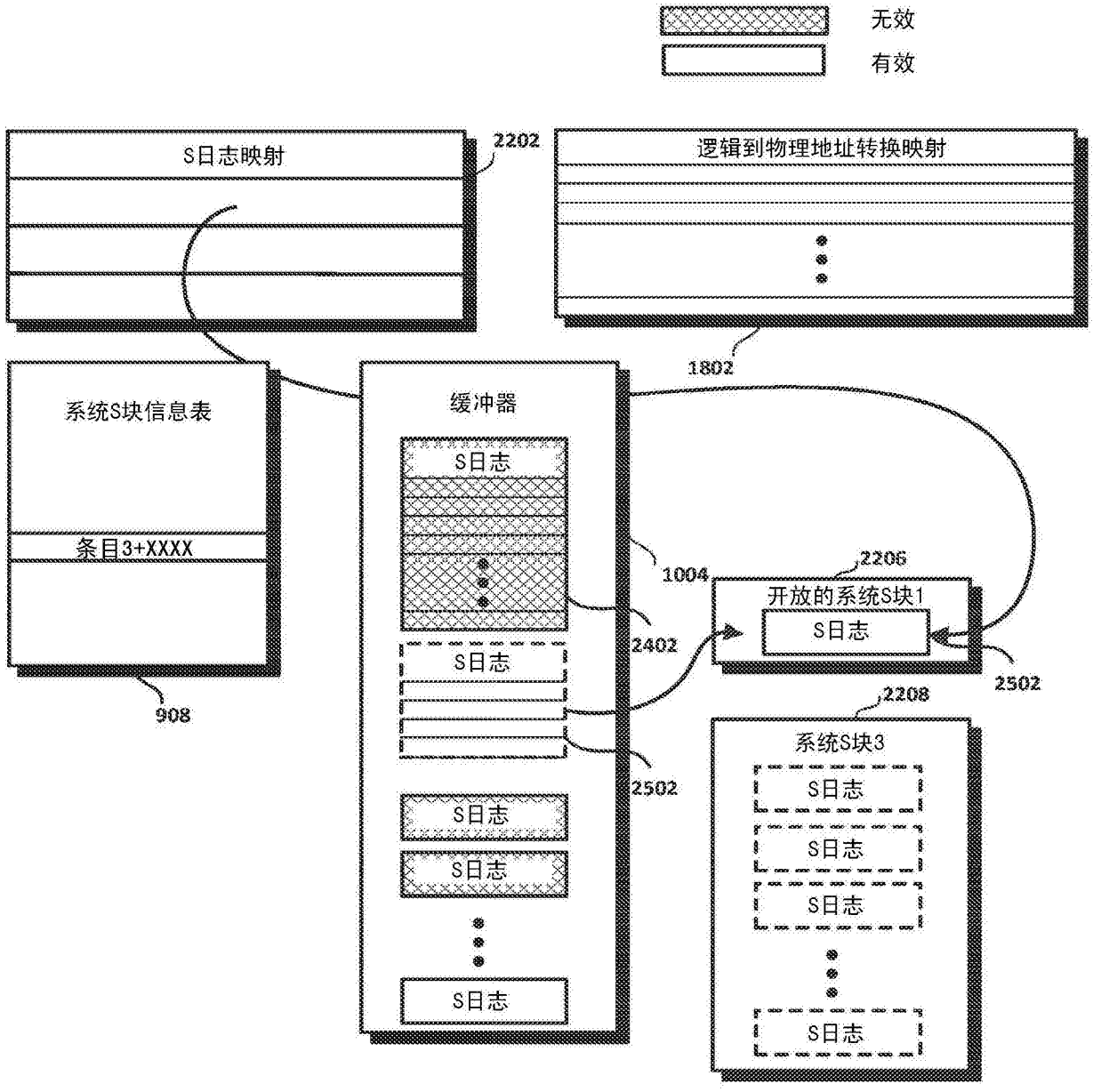

无效
有效
S日志映射
2202
逻辑到物理地址转换映射
1802
系统S块信息表
条目3+XXXX
908
缓冲器
S日志
S日志
S日志
S日志
S日志
1004
2402
2502
2206
开放的系统S块1
S日志
2502
2208
系统S块3
S日志
S日志
S日志
S日志

图 26

Abstract

A data storage device comprises a plurality of non-volatile memory devices storing physical pages, each stored at a predetermined physical location. A controller may be coupled to the memory devices and configured to access data stored in a plurality of logical pages (L-Pages), each associated with an L-Page number that enables the controller to logically reference data stored in the physical pages. A volatile memory may comprise a logical-to-physical address translation map that enables the controller to determine a physical location, within the physical pages, of data stored in each L-Page. The controller may be configured to maintain, in the memory devices, journals defining physical-to-logical correspondences, each journal covering a predetermined range of physical pages and comprising a plurality of entries that associate one or more physical pages to each L-Page. The controller may read the journals upon startup and rebuild the address translation map from the read journals.