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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
3 November 2016 (03.11.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/175762 A1

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

(21) International Application Number:
PCT/US2015/027999

(22) International Filing Date:
28 April 2015 (28.04.2015)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT PACKARD ENTERPRISE DEVELOPMENT LP** [US/US]; 11445 Compaq Center Drive West, Houston, TX 77070 (US).

(72) Inventors: **DENEUI, Nathaniel S**; 11445 Compaq Center Dr W, Houston, Texas 77070 (US). **ORSAK, Keith**; 11445 Compaq Center Dr W, Houston, Texas 7707 (US). **BLACK, Joseph David**; 8138 Vintage Creek Drive, Spring, Texas 77379 (US). **YATES, James Kenneth**; 4911 Dumfries Drive, Houston, Texas 77096 (US).

(74) Agents: **ORTEGA, Arthur** et al.; Hewlett Packard Enterprise, 3404 E. Harmony Road, Mail Stop 79, Fort Collins, CO 80528 (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, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, 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, ST, 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).

Declarations under Rule 4.17:

— *as to the identity of the inventor (Rule 4.17(i))*

[Continued on next page]

(54) Title: STORAGE CACHE MANAGEMENT

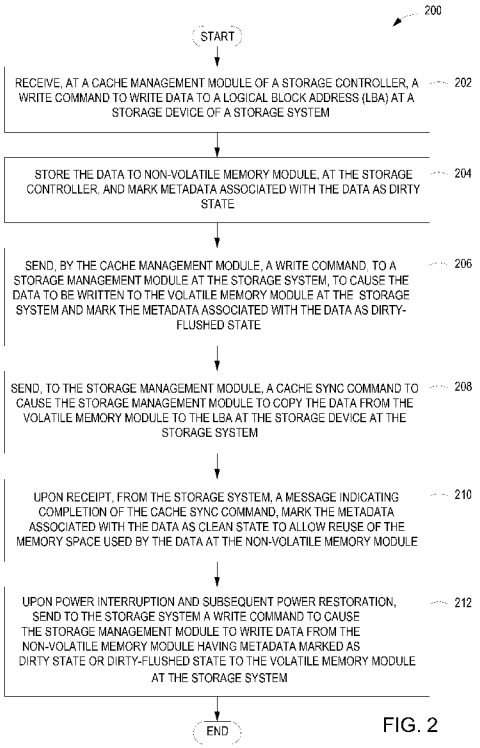

FIG. 2

(57) Abstract: In one example, techniques for storage cache management include receiving a write command to write data to a logical block address (LBA) at a storage device of a storage system, storing the data to the non-volatile memory module and mark metadata associated with the data as Dirty state, sending a write command to write the data to volatile memory module at the storage system and mark the metadata associated with the data as Dirty-Flushed state, sending a cache sync command to copy the data from the volatile memory module to the LBA at the storage device at the storage system, and upon receipt, from the storage system, a message indicating completion of the cache sync command, marking the metadata associated with the data as Clean state to allow reuse of the memory space used by the data at the non-volatile memory module.

WO 2016/175762 A1

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

Published:

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

STORAGE CACHE MANAGEMENT

BACKGROUND

[0001] Computer systems include host computers that communicate with storage systems. The storage systems may include storage devices to store data for later retrieval. The host computers may send commands to the storage systems to have the storage systems store data at the storage devices as well as retrieve data from the storage devices.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] Examples are described in the following detailed description and in reference to the drawings, in which:

[0003] Fig. 1 depicts an example system comprising storage cache management in accordance with an example of the techniques of the present application;

[0004] Fig. 2 depicts an example flow chart of processes for storage cache management in accordance with an example of the techniques of the present application;

[0005] Fig. 3 depicts an example flow diagram of processes for storage cache management in accordance with an example of the techniques of the present application; and

[0006] Fig. 4 depicts an example block diagram showing a non-transitory, computer-readable medium that stores instructions for storage cache management in accordance with an example of the techniques of the present application.

DETAILED DESCRIPTION

[0007] Computer systems include host computers that communicate with storage systems. The storage systems may include storage devices to store data for later retrieval. The host computers may send commands to the storage systems to have the storage systems store data at the storage devices as well as retrieve data from the storage devices. Computer systems may include a storage controller to interface between host computers and storage systems. The storage controller may include non-volatile memory configured as write cache memory. The storage controller may use the write cache memory to construct or build volumes or connect to the storage system in the form of physical disks of the storage devices. The storage controller may use the write cache to redirect write commands to the write cache for buffering Input-Output (IO) processing to help improve write completion, write coalescing, and write sorting. In one example, write

completion may include a condition or situation where the storage controller has stored the data to persistent memory such as non-volatile memory without having committed the data to the storage device (end device) which may improve performance of a host application. In another example, write coalescing may include a condition or situation of combining adjacent Logical Block Address (LBA) data into large flush requests which may reduce the number of IO activity and increase the size of the transfer which may improve the performance of the storage device (end device). In one example, write sorting may include a condition or situation where for certain storage devices, such as Hard Disk Drives (HDDs), IO activity may be sorted relative to LBA location to help minimize seek times and improve performance. The storage device of the storage system (end device) may comprise a volatile memory module configured as write cache memory. The write cache may not be configured to support power loss or interruption which may result in exposing the data written to the write cache to possible loss if the computer system or components of the computer system experiences power loss or interruption. The storage system performance may be improved by enabling or turning on the volatile memory write cache. For example, this may improve storage system write coalescing, storage system write sorting, additional staging area for read-modify-write operations, and enhanced performance for emulation type devices such as HDDs, Shingled Magnetic Recording (SMR) devices, and the like.

[0008] The storage controller may be configured to implement techniques for flushing or writing data to the storage devices such as HDDs which may help reduce the possibility of data loss. This may improve storage system performance by enabling the volatile memory write cache of the storage system and keeping all data in the storage controller cache until a subsequent cache sync command completion message is successfully returned from the storage system. In one example, the storage controller cache may be configured without enabling the storage system write cache. In this case, in operation, the host computer may send or issue a write command to the storage controller to cause the storage controller to write data to a LBA X at the storage system, where X represents any LBA on the storage device. The storage controller buffers data associated the LBA X in the write cache of the storage controller and marks metadata associated with the data for LBA X to Dirty state. At some later time, based on particular criteria such as a sorted list or by time period, the storage controller determines to flush or write the data for LBA X to the storage device. In this case, the storage controller generates and sends a write command to write the data to LBA X at the storage device. At some point, the LBA X operation completes and the storage controller marks the metadata associated with the

data for LBA X in its cache to the Clean state to allow the storage controller to reuse the space used by the data. However, it may be desirable for the storage controller to be able to provide proper management of power loss or interruption.

[0009] To help improve storage performance of storage systems, in one example, techniques are disclosed that provide a two-tier cache configuration where the storage controller may include non-volatile memory module configured as a write cache (Tier 1) and the storage device may include volatile memory module configured as a write cache (Tier 2). In one example, the storage controller may be configured to receive from the host computer write commands to write data to locations such as LBA X at storage device. The storage controller buffers the data for LBA X in the non-volatile memory module configured as write cache and marks the associated metadata for LBA X to the Dirty state. At some later time, based on particular criteria such as a sorted list (e.g., sorting process describe above) or by time period (e.g., stale data timer), the storage controller determines to flush the data for LBA X to the storage device. In this case, the storage controller generates and sends a write command to write the data to LBA X to the storage device. At some point (e.g., upon completion of write to secondary storage or volatile storage), the LBA X operation completes and the storage controller marks the metadata associated with the data for LBA X in its cache to the Dirty-Flushed state. At some future point, the storage controller sends or issues to the storage system a cache sync command to cause the data to be flushed or written from the volatile memory configured as cache to the storage device. Once the storage device completes the write operation, the storage system sends to the storage controller a message indicating completion of the cache sync command. At this point, the storage controller marks the associated metadata for LBA X to Clean state and allowing it to be reused for subsequent write commands from the host computer or other purpose.

[00010] In this manner, the two-tier cache technique allow the end-to-end throughput of the system to improve since now the storage system may be able to employ or implement additional optimization operations using the write cache on the storage system.

[00011] These techniques may allow storage systems to better handle power loss or interruption conditions. Power loss or interruption conditions may be defined as conditions or events such that the components of the system such as those memory components of the storage controller and storage system may no longer be able to maintain data without having to backup data to non-volatile storage. Power restoration conditions may be defined as conditions or events where power is restored to components

of the system such as the storage controller and storage system such as memory so to be able to maintain data without having to backup data to a non-volatile storage For example, in the event of a power loss or interruption, storage controller provides a backup of the data from non-volatile memory and load back to it the volatile memory upon on subsequent power on or power restoration event. In this case, the storage controller may attempt to write to the storage device data corresponding to the LBAs that still have metadata marked Dirty or Dirty-Flushed state. For the storage system, since the loss of power may have caused the data associated with the write command operation to not have been written to the storage device, the storage controller is configured to resend write commands to the storage system for corresponding LBAs to ensure that the data is written or placed on the storage device. In this manner, these techniques may help provide data integrity and reduce any negative impact of data loss that was stored on volatile memory configured as write cache on the storage system that was not committed to the storage device.

[00012] In some examples, the present application discloses techniques which may help increase storage system performance by allowing the storage system to use a volatile memory configured as write cache with a backup stored in a non-volatile memory configured as power loss safe cache managed by the storage controller. In one example, the storage controller may be configured to retain or hold on to data that was flushed from the non-volatile memory configured as write cache until it receives a response or message from the storage system indicating completion of the data written to the storage device. In this manner, this technique may help ensure that the data written to the storage device was successfully completed. In another example, these techniques may enable storage system with volatile memory which may be encounter data loss in power loss conditions to be utilized in a power safe method. In this manner, storage systems with volatile memory configured as write cache may improve performance and endurance.

[00013] In one example, the present application discloses techniques for storage cache management that include a storage controller configured to receive a write command to write data to an LBA at a storage device of a storage system, store the data to non-volatile memory module, and mark metadata associated with the data as Dirty state The storage controller is configured to send a write command to cause the data to be written to the volatile memory module at the storage system and mark the metadata associated with the data as Dirty-Flushed state. The storage controller is configured to send a cache sync command to cause the data from the volatile memory module to be copied to the LBA at the storage device at the storage system. The storage controller is configured to, upon receipt of message indicating completion of the cache sync command,

mark the metadata associated with the data as Clean state to allow reuse of the memory space used by the data at the non-volatile memory module. The storage controller is configured to, upon power interruption and subsequent power restoration, send a write command to cause the data to be written from the non-volatile memory module having metadata marked as Dirty state or Dirty-Flushed state to the volatile memory module at the storage system. In other words, in the event of power loss condition, the data at the volatile-memory module may not have been successfully written to the storage device. The storage controller stills maintains the data in non-volatile memory module that has metadata of the data in the Dirty or Dirty-Flushed state. To help ensure that the data at the volatile-memory module is successfully written to storage device, the storage controller sends write commands to the storage system to resends the data from the non-volatile memory module to the volatile memory module so that the data may be written to the storage device because of the power loss condition the storage system.

[00014] In this manner, in some examples, the present application may provide storage cache management techniques which may help improve the performance of storage systems. For example, these techniques may enable storage systems with volatile memory configured as write cache to increase throughput with reduced risk of data loss in the event of power loss or interruption. In another example, these techniques may allow management of relatively low IO loads and allow storage systems to observe lower latencies for operations that may have blocked storage controller write operations directed to the cache of the storage controllers. In one example, these techniques may allow systems with storage devices with different relatively large physical block size compared to logical block size to potentially utilize additional memory for read-modify-write operations. In one example, this may relate to devices with larger physical block size relative to logical block size. In order to access large blocks, the physical blocks may first be read, modified on the logical block level then re-written to a physical block. In another example, the techniques may allow storage systems with storage devices to perform bookkeeping operations and reduce impacts due to write buffering being utilized.

[00015] Fig. 1 depicts an example system 100 for storage cache management in accordance with an example of the techniques of the present application. The system 100 includes a storage controller 104 configured to communicate with a host computer 102 and communicate with a storage system 110.

[00016] The host computer 102 can be any electronic device configured to perform data processing. The host computer 102 may be implemented in hardware, software, or a combination thereof. The host computer 102 is configured to communicate with storage

controller 104. For example, host computer 102 may send to storage controller 104 write commands to write data to locations such as LBAs at storage device 116 of storage system 110. In another example, host computer 102 may send to storage controller 104 read commands to read data from locations such as LBAs at storage device 116 of storage system 110. The techniques of the present application are described in the context to read and write data or groups of data such as data blocks at LBA address locations at a storage device but it should be understood that the techniques may be applicable to other location address techniques such as physical address blocks, Logical Unit Number (LUN) and the like.

[00017] The storage controller 104 includes a cache management module 106 and a non-volatile memory module 108. The storage controller 104 may be any electronic device configured to perform data processing. The storage controller 104 and the components of the storage controller such as cache management module 106 may be implemented in hardware, software, or a combination thereof.

[00018] The cache management module 106 may be configured to manage communications between host computer 102 and storage system 110. The cache management module 106 may be configured to keep track of write commands from host computer 102 using metadata and state logic associated with the different states of the operation associated with the write commands. For example, as explained below in further detail, the cache management module 106 is configured to mark metadata associated with the data to be written to storage device 116 in accordance with three states: Dirty state, Dirty-Flushed State, Clean state. The Dirty state represents a condition or state where storage controller 104 has written the data, received from a write command from host computer 102, to non-volatile memory module 108. The Dirty-Flushed state represents a condition where storage controller 104 has caused storage system 110 to copy or write the data from non-volatile memory module 108 to volatile memory module 114. The Clean state represents a condition where storage controller 104 has caused storage system 110 to write the data from volatile memory module 114 to storage device 116.

[00019] In one example, in operation, cache management module 106 may receive write commands to write data to LBAs at storage device 116 of storage system 110. The cache management module 106 may store the data to non-volatile memory module 108 and mark metadata associated with the data as Dirty state. The cache management module 106 may send a write command to storage management module 112 to cause the data to be written to volatile memory module 114 and mark the metadata associated with the data as Dirty-Flushed state. The cache management module 106 may send, to storage

management module 112, a cache sync command to cause the storage management module to copy the data from volatile memory module 114 to the LBA at storage device 116 at storage system 110. In one example, the cache sync commands may be sent to the storage system at various times or conditions such as when the cache is full, when a stale timer has expired, when the device optimal write boundary is reached and the like. The cache management module 106 upon receipt, from storage system 110, a message indicating completion of the cache sync command, may mark the metadata associated with the data as Clean state to allow reuse of the memory space used by the data at non-volatile memory module 108. The cache management module 106, upon power interruption and subsequent power restoration, send to storage system 110 a write command to cause storage management module 112 to write data from non-volatile memory module 108 having metadata marked as Dirty state or Dirty-Flushed state to volatile memory module 114 at the storage system.

[00020] In other words, in the event of power loss condition, the data at volatile-memory module 114 may not have been successfully written to storage device 116. The storage controller 104 stills maintains the data in non-volatile memory module 108 that has metadata of the data in the Dirty or Dirty-Flushed state. To help ensure that the data at volatile-memory module 114 is successfully written to storage device 116, storage controller 104 sends write commands to storage system 110 to resend the data from non-volatile memory module 108 to volatile memory module 114 so that the data may be written to the storage device because of the power loss condition of the system.

[00021] The cache management module 106 may be configured to manage the timing of the generation and transmission of cache sync commands or operations directed to storage system 110. For example, cache management module 106 may be configured to send cache sync commands to storage system 110 based on a pattern of previous cache sync commands. In one example, the pattern of previous cache sync commands may be based on a measurement or metrics of storage device 116 performance. In another example, cache management module 106 may be configured to send cache sync commands to storage system 110 based on hit rate of the non-volatile memory module 108. For example, the hit rate of the non-volatile memory 108 may be based on amount of data stuck in memory such as a relatively large amount of data stuck in the Dirty-Flushed metadata state which may suggest the storage controller may not enough memory space for future operations. In another example, cache management module 106 may be configured to send cache sync commands to storage system 110 based on a pattern of write commands sent to storage system 110. For example, cache management module

106 may encounter relatively high sequential patterns which may result in higher throughput and write coalescing thereby generate cache sync commands at a higher rate to lower the amount of data which may have metadata marked as the Dirty-Flushed state.

[00022] The storage system 110 includes storage management module 112, a volatile memory module 114, and storage device 116. The storage system 110 may be any electronic device configured to perform data processing. The storage system 110 and the components of the storage system may be implemented in hardware, software, or a combination thereof.

[00023] In one example, storage management module 112 may be configured to manage communication with storage controller 104. For example, storage management module 112 may be configured to receive write commands from storage controller 104 to write data to volatile memory module 114 of storage system 110. In this case, storage management module 112 may mark metadata associated with the data as the Dirty state. In another example, storage management module 112 may be configured to receive cache sync commands from storage controller 104 to write data from volatile memory module 114 to storage device 116. Upon successful completion of cache sync command, storage management module 112 may respond to storage controller 104 with a message indicating completion of the cache sync command. In this case, storage management module 112 may mark metadata associated with the data as the Clean state. In this manner, storage controller may now be able to reuse the memory space used by the data at non-volatile memory module 108.

[00024] The storage controller 104 is configured with non-volatile memory module 108 which comprise any electronic means for storing data for later retrieval by storage controller 104 even after power loss condition. The non-volatile memory module 108 may be defined as memory that may be configured to store data or information and be later retrieved even after experiencing a power loss condition. That is, non-volatile memory module 108 may retain data even without power. Some examples of non-volatile memory module 108 may include Flash memory, magnetic type computer devices, such as hard disk drives and the like. The storage controller 104 may configure non-volatile memory module 108 to provide cache memory space at the storage controller separate from volatile memory module 114 to provide cache memory space at the storage system. The cache memory space may be defined as memory space to store data that is frequently accessed by storage controller 104. In one example, the cache memory space way include configuration of cache memory as write cache memory such that data frequently written to storage is stored to the write cache memory. In another example, the cache

memory space way include configuration of the cache memory as read cache memory such that data frequently read from storage is stored to the read cache memory. The storage controller 104 may configure cache memory space as write cache, read cache or a combination thereof.

[00025] The storage system 110 may be configured with volatile memory module 114 which may comprise any electronic means for storing data for later retrieval by storage system 110 but is not able to maintain the data after a power loss condition which may cover volatile and non-volatile conditions or situations. The volatile memory module 114 may be defined as memory that may store data or information but requiring power to maintain the stored information. That is, the stored information is lost if module 114 experiences a power loss condition. Examples of volatile memory module 114 may include general Random Access Memory (RAM) and the like. The storage system 110 may configure volatile memory module 114 to provide cache memory space at the storage system separate from non-volatile memory module 108 to provide cache memory space at the storage controller 104. The storage system 110 may configure cache memory space as write cache, read cache or a combination thereof.

[00026] In this manner, in some examples, the techniques of the present application may provide storage cache management techniques to help improve the performance of storage systems.

[00027] The system 100 of Fig. 1 shows an example storage controller 104 and should be understood that other configurations may be employed to practice the techniques of the present application. For example, system 100 may be configured to communicate with a plurality of storage controllers 104 and with a plurality of host computers 102 and with a plurality of storage systems 110. The components of system 100 may be implemented in hardware, software or a combination thereof. For example, the functionality of the components of system 100 may be implemented using Personal Computers (PCs), server computers, tablet computers, mobile computers and the like. The storage controller 104, host computer 102 and storage system 110 may communicate using any communications means such as Fibre Channel, Ethernet and the like.

[00028] Fig. 2 depicts an example flow chart 200 of a process for storage cache management in accordance with an example of the techniques of the present application. To illustrate operation, it may be assumed that storage controller 104 is configured to communicate with host computer 102 and storage system 110 as described herein. It may be assumed that storage controller storage controller 104 includes cache management module 106 to implement the functionality described herein.

[00029] It should be understood the process depicted in Fig. 2 represents generalized illustrations, and that other processes may be added or existing processes may be removed, modified, or rearranged without departing from the scope and spirit of the present application. In addition, it should be understood that the processes may represent instructions stored on a processor-readable storage medium that, when executed, may cause a processor to respond, to perform actions, to change states, and/or to make decisions. Alternatively, the processes may represent functions and/or actions performed by functionally equivalent circuits like analog circuits, digital signal processing circuits, application specific integrated circuits (ASICs), or other hardware components associated with the system. Furthermore, the flow charts are not intended to limit the implementation of the present application, but rather the flow charts illustrate functional information to design/fabricate circuits, generate software, or use a combination of hardware and software to perform the illustrated processes.

[00030] The process 200 may begin at block 202, where storage controller receives 104 a write command to write data to an LBA at storage device 116 of storage system 110. In one example, cache management module 106 receives a write command from host computer 102. Processing proceeds to block 204.

[00031] At block 204, storage controller 104 stores the data to non-volatile memory module 108 and marks metadata associated with the data as Dirty state. In one example, storage controller 104 may send a write completion message to the host computer 102 after the metadata of the data is marked as Dirty state. Processing proceeds to block 206.

[00032] At block 206, storage controller 104 sends a write command to storage management module 112 to cause the data to be written to volatile memory module 114 and marks the metadata associated with the data as Dirty-Flushed state. Processing proceeds to block 208.

[00033] At block 208, storage controller 104 sends, to storage management module 112, a cache sync command to cause the storage management module to copy the data from volatile memory module 114 to the LBA at storage device 116 at storage system 110. Processing proceeds to block 210.

[00034] At block 210, storage controller 104, upon receipt, from storage system 110, a message indicating completion of the cache sync command, marks the metadata associated with the data as Clean state to allow reuse of the memory space used by the data at non-volatile memory module 108. Processing proceeds to block 212.

[00035] At block 212, storage controller 104, upon power interruption and subsequent power restoration, sends to storage system 110 a write command to cause

storage management module 112 to write data from non-volatile memory module 108 having metadata marked as Dirty state or Dirty-Flushed state to volatile memory module 114 at the storage system. In other words, in the event of power loss condition, the data at volatile-memory module 114 may not have been successfully written to storage device 116. The storage controller 104 stills maintains the data in non-volatile memory module 108 as in the Dirty or Dirty-Flushed state. To help ensure that the data at volatile-memory module 114 is successfully written to storage device 116, storage controller 104 sends write commands to storage system 100 to resend the data from non-volatile memory module 108 to volatile memory module 114 so that the data may be written to storage device 1116 because of the power loss condition of the system. Processing proceeds to End block. In another example, processing may proceed to back block 202 to process further write commands from host computer 102.

[00036] In this manner, in some examples, the techniques of the present application may provide storage cache management techniques to help improve the performance of storage systems.

[00037] The process 200 of Fig. 2 shows an example process and it should be understood that other configurations can be employed to practice the techniques of the present application. For example, process 200 may be configured to communicate with a plurality of storage devices.

[00038] Fig. 3 depicts an example flow diagram 300 of processes for storage cache management in accordance with an example of the techniques of the present application. To illustrate operation, it may be assumed that storage controller 104 is configured to communicate with host computer 102 and storage system 110 as described herein. It may be assumed that storage controller 104 includes cache management module 106 to implement the functionality described herein.

[00039] The process 300 may begin at block 301, where storage controller 104 receives, from host computer 102, a write command to write data to an LBA at storage device 116 of storage system 110. Processing proceeds to block 302.

[00040] At block 302, storage controller 104 writes or stores the data to non-volatile memory module 108 and marks metadata associated with the data as Dirty state. Processing proceeds to block 303.

[00041] At block 303, storage controller 104 sends to host computer 102 a response or message indicating completion of the write command. Processing proceeds to block 304.

[00042] At block 304, storage controller 104 flushes the write command or request so to cause the data stored at non-volatile memory module 108 to be copied or written to volatile memory module 114. In one example, cache management module 106 sends, to storage system 110, a write command to cause the storage system to write the data from non-volatile memory module 108 to volatile memory module 114 and mark the metadata associated with the data as Dirty-Flushed state. Processing proceeds to block 305.

[00043] At block 305, storage system 110 writes the data to volatile memory module 114 and marks metadata associated from Clean state to Dirty state. In one example, after the data is written to volatile memory module 114 at storage system 110, storage system marks the metadata at the storage system associated with the data to the Dirty state. Processing proceeds to block 306.

[00044] At block 306, storage controller 104 sends a cache sync command to storage system 110. In one example, storage management module 112 sends a cache sync command to cause storage management module 112 to copy the data from volatile memory module 114 to the LBA at storage device 116 at storage system 110. In another example, storage controller 104 or cache management module 106 may send the cache sync command to storage system 110 based on at least one of a pattern of previous cache sync commands, hit rate of non-volatile memory module 108, and pattern of write commands sent to storage system 110. Processing proceeds to block 307.

[00045] At block 307, storage system 110 writes the data from volatile memory module 114 to the LBA at storage device 116. Processing proceeds to block 308.

[00046] At block 308, storage system 110 marks metadata associated with the data written to the LBA from the Dirty state to Clean state. In one example, after storage system 110 writes the data from the volatile memory module 114 to storage device 116, the storage system marks the metadata at storage system 110 associated with the data to the Clean state. Processing proceeds to block 309.

[00047] At block 309, storage system 110 sends a message or response to storage controller 104 indicating completion of the cache sync command that was previously sent by storage controller 104 at block 306. Processing proceeds to block 310.

[00048] At block 310, storage controller 104 marks metadata associated with data from Dirty-Flushed state to Clean state. In this manner, marking the metadata associated with the data as Clean state may allow reuse of the memory space used by the data at non-volatile memory module 108. In one example, processing may proceed back to block 301 to continue processing other write commands from host computer 102.

[00049] The storage controller 104 may be configured to perform cache storage functions related to power loss or interruption and then power restoration. In one example, upon power interruption and subsequent power restoration conditions, storage controller 104 may send to storage system 110 write commands to cause storage management module 112 to write data from non-volatile memory module 108 having metadata marked as Dirty state or Dirty-Flushed state to volatile memory module 114 at the storage system. In another example, upon power loss or interruption, storage controller 104 may backup the data and restore the data from non-volatile memory module upon power restoration. In another example, in the event of power loss or interruption during process above between blocks 302 and 310, storage controller 104 may cause data at non-volatile memory module 108 having metadata marked as Dirty-Flushed state to be flushed or written to storage system 110. In this case, storage controller 104 may perform the process of blocks 304 through 310 to help ensure data integrity of the data at storage system 110.

[00050] In another example, these techniques may help ensure data integrity of the data at storage system 110. For example, in the event of power loss condition, the data at volatile-memory module 114 may not have been successfully written to storage device 116. The storage controller 104 stills maintains the data in non-volatile memory module 108 which has metadata marked as in the Dirty or Dirty-Flushed state. To help ensure that the data at volatile-memory module 114 is successfully written to storage device 116, storage controller 104 resends write commands to storage system 110 cause the data from non-volatile memory module 108 to be written to volatile memory module 114 so that the data may be written to the storage device because of the power loss condition of the system.

[00051] The storage controller 104 may be configured to handle overlapping or multiple write commands from host computer 102. In one example, host computer 102 may send to storage controller 104 a second write command with data that overlaps with data in non-volatile memory module 108 that has metadata associated with the data marked as in Dirty-Flushed state. In this case, storage controller 104 marks the metadata associated with the data marked as Dirty state and performs the process above of blocks 306 through 308.

[00052] In this manner, in some examples, the techniques of the present application may provide storage cache management techniques to help improve the performance of storage systems.

[00053] The process 300 of Fig. 3 shows an example process and it should be understood that other configurations can be employed to practice the techniques of the present application. For example, process 300 may be configured to communicate with a plurality of storage devices.

[00054] Fig. 4 is an example block diagram showing a non-transitory, computer-readable medium that stores code for operation in accordance with an example of the techniques of the present application. The non-transitory, computer-readable medium is generally referred to by the reference number 400 and may be included in the system in relation to Fig. 1. The non-transitory, computer-readable medium 400 may correspond to any typical storage device that stores computer-implemented instructions, such as programming code or the like. For example, the non-transitory, computer-readable medium 400 may include one or more of a non-volatile memory, a volatile memory, and/or one or more storage devices. Examples of non-volatile memory include, but are not limited to, electrically erasable programmable read only memory (EEPROM) and read only memory (ROM). Examples of volatile memory include, but are not limited to, static random access memory (SRAM), and dynamic random access memory (DRAM). Examples of storage devices include, but are not limited to, hard disk drives, compact disc drives, digital versatile disc drives, optical drives, and flash memory devices.

[00055] A processor 402 generally retrieves and executes the instructions stored in the non-transitory, computer-readable medium 400 to operate the present techniques in accordance with an example. In an example, the tangible, machine-readable medium 400 can be accessed by the processor 402 over a bus 404. A first region 406 of the non-transitory, computer-readable medium 400 may include cache management module 106 functionality as described herein. The cache management module 106 functionality may be implemented in hardware, software or a combination thereof.

[00056] For example, block 408 provides store data instructions which may include instructions to store data to non-volatile memory module 108 and mark metadata associated with the data as Dirty state, as described herein.

[00057] For example, block 410 provides send write command instructions which may include instructions to send a write command to write the data to volatile memory module 114 at storage system 110 and mark the metadata associated with the data as Dirty-Flushed state, as described herein.

[00058] For example, block 412 provides send cache sync command instructions which may include instructions to send a cache sync command to copy the data from

volatile memory module 114 to an LBA at storage device 116 at storage system 110, as described herein.

[00059] For example block 414 provides instructions to mark metadata of the data instructions which may include instructions to mark the metadata associated with the data as Clean state to allow reuse of the memory space used by the data at non-volatile memory module 108, as described herein.

[00060] For example block 416 provides instructions to send write command instructions which may include instructions to, upon power interruption and subsequent power restoration, storage controller 104 sends a write command to write data having metadata marked as Dirty state or Dirty-Flushed state, as described herein.

[00061] Although shown as contiguous blocks, the software components can be stored in any order or configuration. For example, if the non-transitory, computer-readable medium 400 is a hard drive, the software components can be stored in non-contiguous, or even overlapping, sectors

[00062] The foregoing describes a novel and previously unforeseen approach for storage cache management. While the above application has been shown and described with reference to the foregoing examples, it should be understood that other forms, details, and implementations may be made without departing from the spirit and scope of this application.

[00063]

WHAT IS CLAIMED IS:

1. A storage controller comprising:

a non-volatile memory module to provide memory space to store data; and

a cache management module to:

receive, from a host computer, a write command to write data to a logical block address (LBA) at a storage device of a storage system,

store the data to the non-volatile memory module and mark metadata associated with the data as Dirty state,

send, to the storage system, a write command to write the data to volatile memory module at the storage system and mark the metadata associated with the data as Dirty-Flushed state,

send, to the storage system, a cache sync command to copy the data from the volatile memory module to the LBA at the storage device at the storage system,

receive a message indicating completion of the cache sync command and mark the metadata associated with the data as Clean state to allow reuse of the memory space used by the data at the non-volatile memory module; and

upon power interruption and subsequent power restoration, send to the storage system a write command to write data having metadata marked as Dirty state or Dirty-Flushed state.

2. The storage controller of claim 1, wherein the cache management module to send a write completion message to the host after the metadata of the data is marked as Dirty state.

3. The storage controller of claim 1, wherein after the data is written to volatile memory module at the storage system, metadata at the storage system associated with the data is marked as Dirty state, and after the data is written from the volatile memory module to the storage device, the metadata at the storage system associated with the data is marked as Clean state.

4. The storage controller of claim 1, wherein the cache management module to send the cache sync command to the storage system based on at least one of a pattern of

previous cache sync commands, hit rate of the non-volatile memory, and pattern of write commands sent to the storage system.

5. The storage controller of claim 1, wherein the non-volatile memory module to provide cache memory space at the storage controller separate from the volatile memory module to provide cache memory space at the storage system.

6. A method of storage cache management comprising:

receiving, at a cache management module of a storage controller, a write command to write data to a logical block address (LBA) at a storage device of a storage system;

storing the data to non-volatile memory module, at the storage controller, and marking metadata associated with the data as Dirty state;

sending, by the cache management module, a write command, to a storage management module at the storage system, to cause the data to be written to the volatile memory module at the storage system and marking the metadata associated with the data as Dirty-Flushed state;

sending, to the storage management module, a cache sync command to cause the storage management module to copy the data from the volatile memory module to the LBA at the storage device at the storage system;

upon receipt, from the storage system, a message indicating completion of the cache sync command, marking the metadata associated with the data as Clean state to allow reuse of the memory space used by the data at the non-volatile memory module; and

upon power interruption and subsequent power restoration, sending to the storage system a write command to cause the storage management module to write data from the non-volatile memory module having metadata marked as Dirty state or Dirty-Flushed state to the volatile memory module at the storage system.

7. The method of claim 6, further comprising sending a write completion message to the host after the metadata of the data is marked as Dirty state.

8. The method of claim 6, wherein after the data is written to volatile memory module at the storage system, metadata at the storage system associated with the data is marked as Dirty state, and after the data is written from the volatile memory module to the

storage device, the metadata at the storage system associated with the data is marked as Clean state.

9. The method of claim 6, wherein sending the cache sync command to the storage system is based on at least one of a pattern of previous cache sync commands, hit rate of the non-volatile memory, and pattern of write commands sent to the storage system.

10. The method of claim 6, further comprising providing the non-volatile memory module as cache memory space at the storage controller separate from providing the volatile memory module as cache memory space at the storage system.

11. A non-transitory computer-readable medium having computer executable instructions stored thereon for storage cache management, the instructions are executable by a processor to:

store data to non-volatile memory module and mark metadata associated with the data as Dirty state;

send a write command to write the data to volatile memory module at the storage system and mark the metadata associated with the data as Dirty-Flushed state;

send a cache sync command to copy the data from the volatile memory module to a logical block address (LBA) at the storage device at the storage system;

receive a message indicating completion of the cache sync command and mark the metadata associated with the data as Clean state to allow reuse of the memory space used by the data at the non-volatile memory module; and

upon power interruption and subsequent power restoration, send a write command to write data having metadata marked as Dirty state or Dirty-Flushed state.

12. The non-transitory computer-readable medium of claim 11, further comprising instructions that if executed cause a processor to send a write completion message to the host after the metadata of the data is marked as Dirty state.

13. The non-transitory computer-readable medium of claim 11, further comprising instructions that if executed cause a processor to: wherein after the data is written to volatile memory module at the storage system, metadata at the storage system associated with the data is marked as Dirty state, and after the data is written from the

volatile memory module to the storage device, the metadata at the storage system associated with the data is marked as Clean state.

14. The non-transitory computer-readable medium of claim 11 further comprising instructions that if executed cause a processor to: send the cache sync command to the storage system is based on at least one of a pattern of previous cache sync commands, hit rate of the non-volatile memory, and pattern of write commands sent to the storage system.

15. The non-transitory computer-readable medium of claim 11 further comprising instructions that if executed cause a processor to: provide the non-volatile memory module as cache memory space at the storage controller separate from to provide the volatile memory module as cache memory space at the storage system.

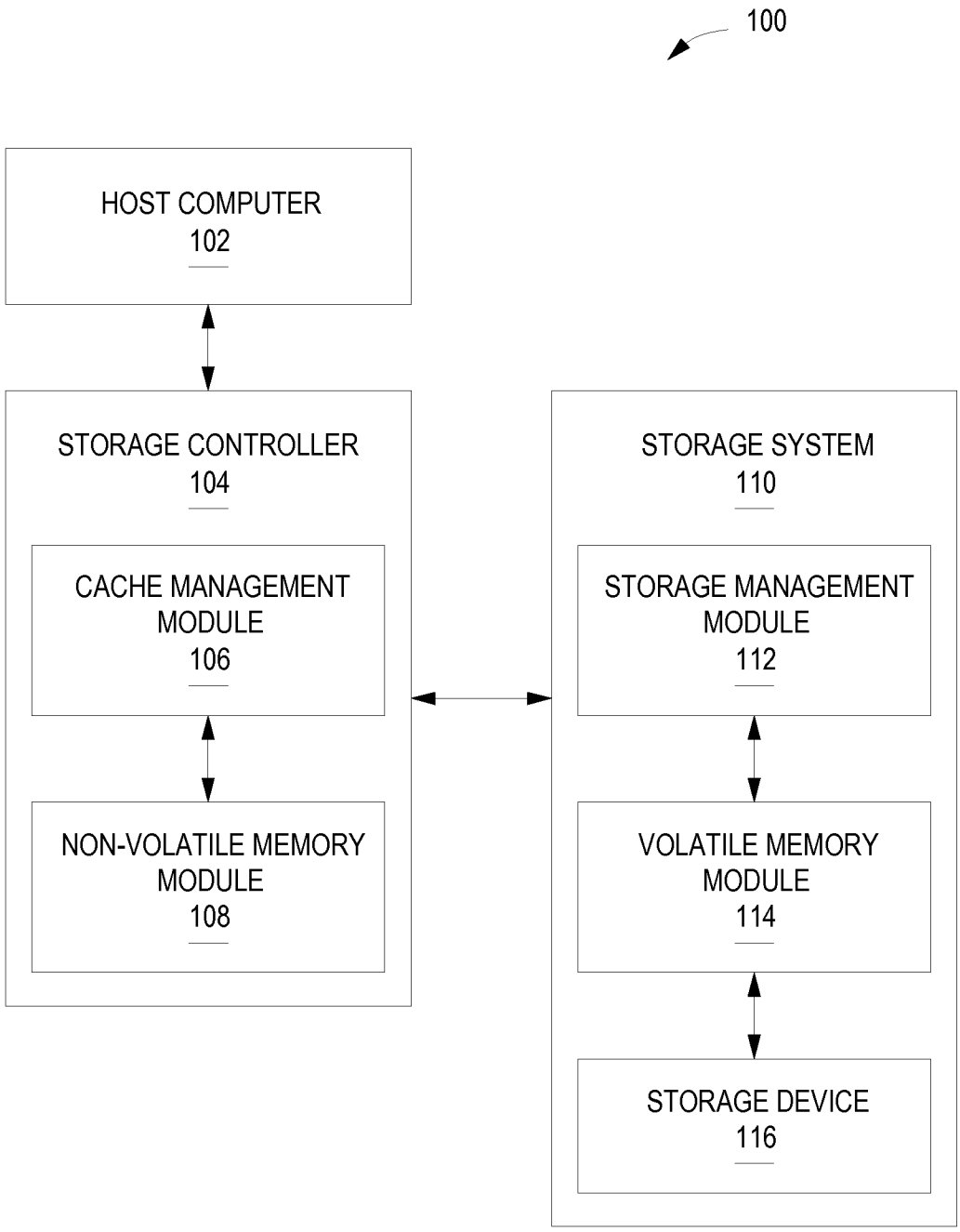
100
HOST COMPUTER
102
STORAGE CONTROLLER
104
CACHE MANAGEMENT MODULE
106
NON-VOLATILE MEMORY MODULE
108
STORAGE SYSTEM
110
STORAGE MANAGEMENT MODULE
112
VOLATILE MEMORY MODULE
114
STORAGE DEVICE
116

FIG. 1

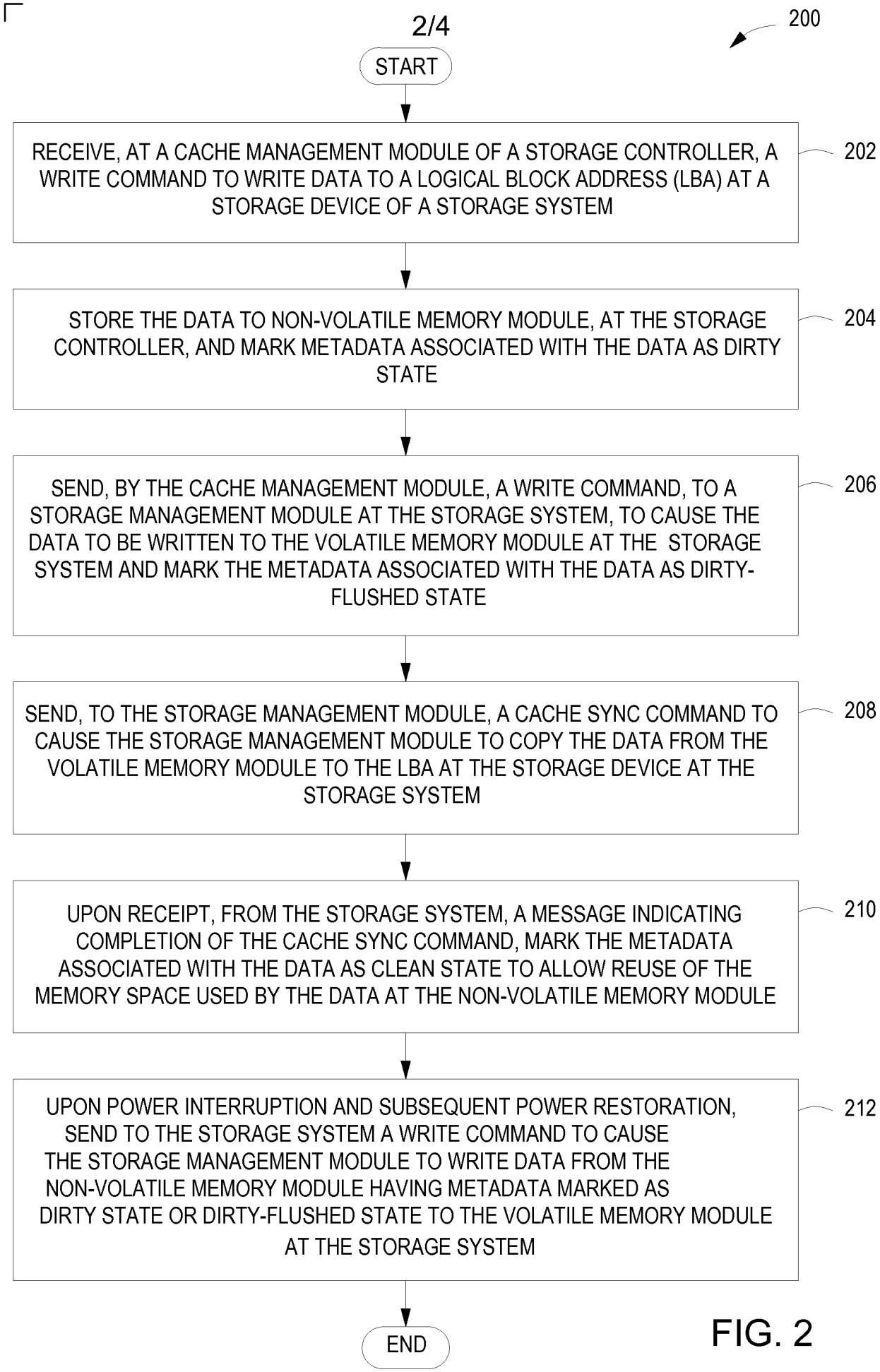
200
START
RECEIVE, AT A CACHE MANAGEMENT MODULE OF A STORAGE CONTROLLER, A WRITE COMMAND TO WRITE DATA TO A LOGICAL BLOCK ADDRESS (LBA) AT A STORAGE DEVICE OF A STORAGE SYSTEM
202
STORE THE DATA TO NON-VOLATILE MEMORY MODULE, AT THE STORAGE CONTROLLER, AND MARK METADATA ASSOCIATED WITH THE DATA AS DIRTY STATE
204
SEND, BY THE CACHE MANAGEMENT MODULE, A WRITE COMMAND, TO A STORAGE MANAGEMENT MODULE AT THE STORAGE SYSTEM, TO CAUSE THE DATA TO BE WRITTEN TO THE VOLATILE MEMORY MODULE AT THE STORAGE SYSTEM AND MARK THE METADATA ASSOCIATED WITH THE DATA AS DIRTY-FLUSHED STATE
206
SEND, TO THE STORAGE MANAGEMENT MODULE, A CACHE SYNC COMMAND TO CAUSE THE STORAGE MANAGEMENT MODULE TO COPY THE DATA FROM THE VOLATILE MEMORY MODULE TO THE LBA AT THE STORAGE DEVICE AT THE STORAGE SYSTEM
208
UPON RECEIPT, FROM THE STORAGE SYSTEM, A MESSAGE INDICATING COMPLETION OF THE CACHE SYNC COMMAND, MARK THE METADATA ASSOCIATED WITH THE DATA AS CLEAN STATE TO ALLOW REUSE OF THE MEMORY SPACE USED BY THE DATA AT THE NON-VOLATILE MEMORY MODULE
210
UPON POWER INTERRUPTION AND SUBSEQUENT POWER RESTORATION, SEND TO THE STORAGE SYSTEM A WRITE COMMAND TO CAUSE THE STORAGE MANAGEMENT MODULE TO WRITE DATA FROM THE NON-VOLATILE MEMORY MODULE HAVING METADATA MARKED AS DIRTY STATE OR DIRTY-FLUSHED STATE TO THE VOLATILE MEMORY MODULE AT THE STORAGE SYSTEM
212
END

FIG. 2

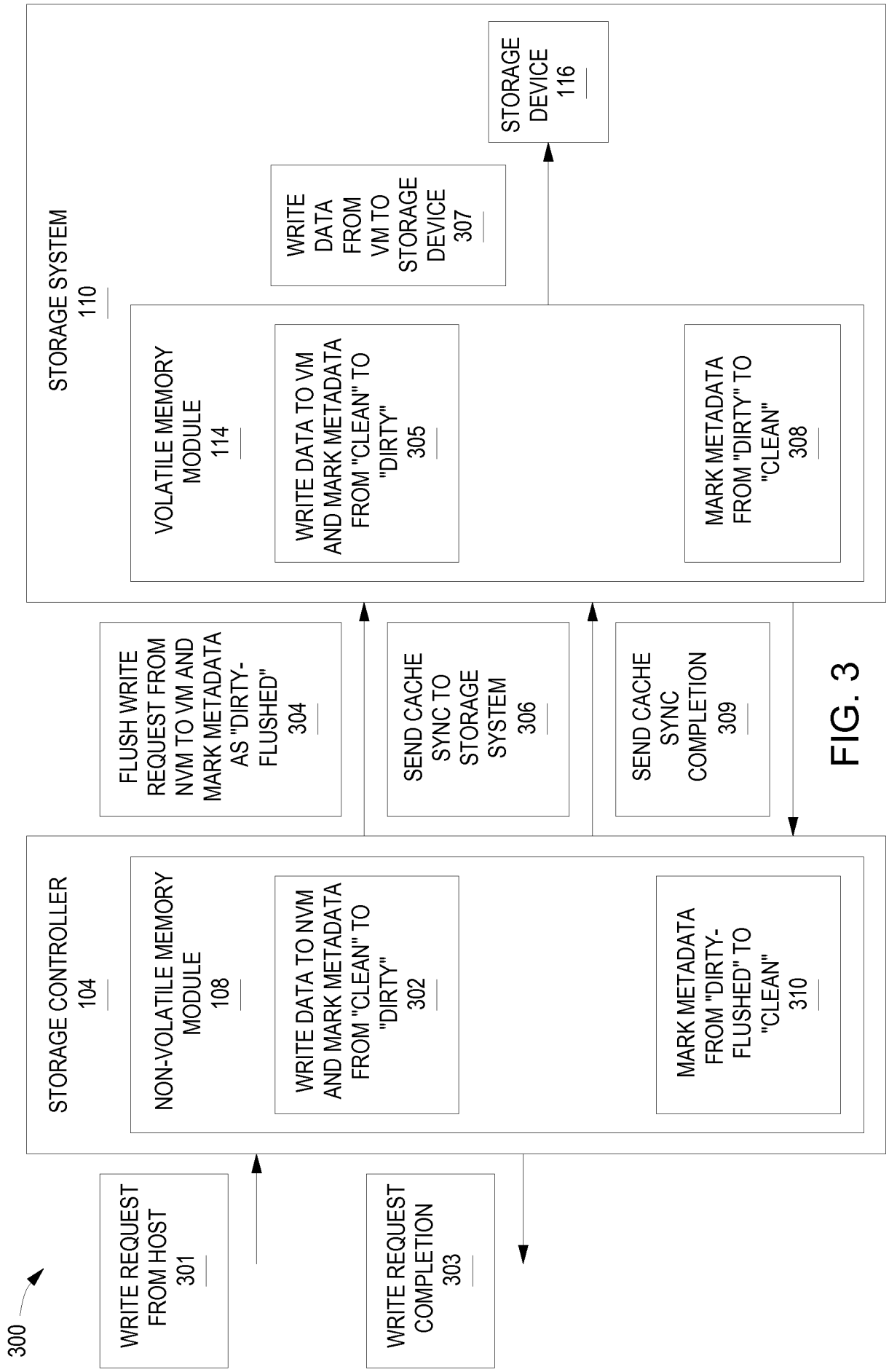
300
WRITE REQUEST FROM HOST 301
WRITE REQUEST COMPLETION 303
STORAGE CONTROLLER 104
NON-VOLATILE MEMORY MODULE 108
WRITE DATA TO NVM AND MARK METADATA FROM "CLEAN" TO "DIRTY" 302
MARK METADATA FROM "DIRTY-FLUSHED" TO "CLEAN" 310
FLUSH WRITE REQUEST FROM NVM TO VM AND MARK METADATA AS "DIRTY-FLUSHED" 304
SEND CACHE SYNC TO STORAGE SYSTEM 306
SEND CACHE SYNC COMPLETION 309
STORAGE SYSTEM 110
VOLATILE MEMORY MODULE 114
WRITE DATA TO VM AND MARK METADATA FROM "CLEAN" TO "DIRTY" 305
MARK METADATA FROM "DIRTY" TO "CLEAN" 308
WRITE DATA FROM VM TO STORAGE DEVICE 307
STORAGE DEVICE 116

FIG. 3

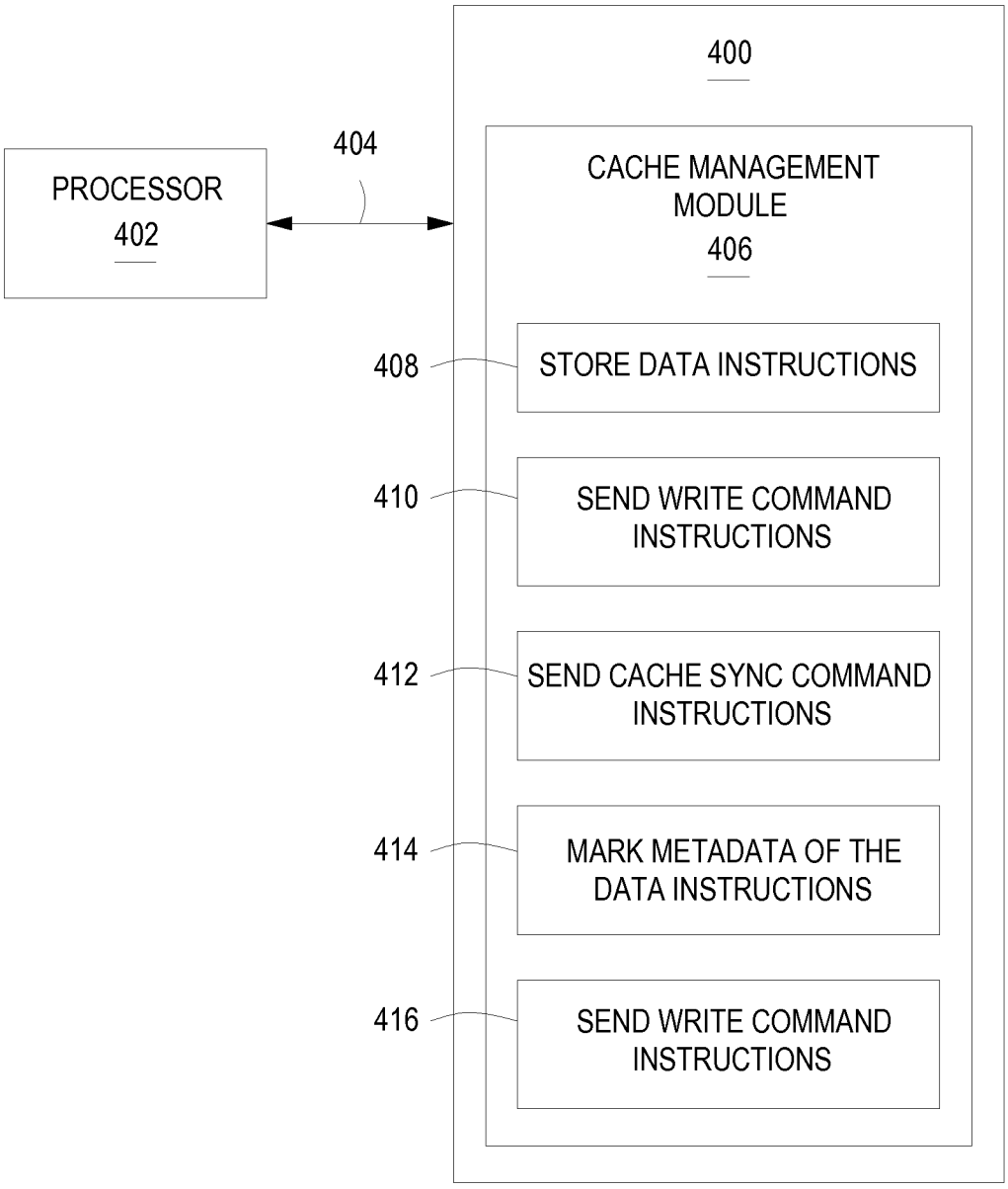
400
404
PROCESSOR
402
CACHE MANAGEMENT
MODULE
406
408
STORE DATA INSTRUCTIONS
410
SEND WRITE COMMAND
INSTRUCTIONS
412
SEND CACHE SYNC COMMAND
INSTRUCTIONS
414
MARK METADATA OF THE
DATA INSTRUCTIONS
416
SEND WRITE COMMAND
INSTRUCTIONS

FIG. 4

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2015/027999

A. CLASSIFICATION OF SUBJECT MATTER

G06F 12/08(2006.01)i, G06F 12/06(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 12/08; G06F 12/12; G06F 3/06; G06F 12/16; G06F 12/00; G06F 13/14; G06F 12/06

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:storage controller, non-volatile memory module, cache management module, dirty state, dirty-flushed state, clean state, logical block address, power interruption

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2013-0297880 A1 (FUSION-IO, INC.) 07 November 2013 See paragraphs [0041], [0051], [0061], [0174], [0215], [0246], [0250], [0319]; and figure 1.	1-15
Y	US 2015-0012690 A1 (ROLANDO H. BRUCE et al.) 08 January 2015 See paragraphs [0007], [0055], [0152], [0158], [0195]; and figure 1.	1-15
A	US 2010-0180065 A1 (JACOB CHERIAN et al.) 15 July 2010 See paragraph [0030]; and figure 2.	1-15
A	WO 2013-095465 A1 (INTEL CORPORATION) 27 June 2013 See paragraph [0013]; and figure 2.	1-15
A	US 2008-0104344 A1 (NORIO SHIMOZONO et al.) 01 May 2008 See paragraph [0058]; and figure 19.	1-15

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

☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier 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
27 January 2016 (27.01.2016)	**01 February 2016 (01.02.2016)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 35208, Republic of Korea
Facsimile No. +82-42-472-7140

Authorized officer
BYUN, Sung Cheal
Telephone No. +82-42-481-8262

Form PCT/ISA/210 (second sheet) (January 2015)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2015/027999

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2013-0297880 A1	07/11/2013	US 2012-210041 A1	16/08/2012
		US 8489817 B2	16/07/2013
		US 8756375 B2	17/06/2014
US 2015-0012690 A1	08/01/2015	None	
US 2010-0180065 A1	15/07/2010	US 9003118 B2	07/04/2015
WO 2013-095465 A1	27/06/2013	CN 103999067 A	20/08/2014
		US 2013-0304978 A1	14/11/2013
US 2008-0104344 A1	01/05/2008	JP 04437489 B2	24/03/2010
		JP 2008-108026 A	08/05/2008
		US 7613877 B2	03/11/2009