ARRAY CONTROLLER, SOLID STATE DISK, AND METHOD FOR CONTROLLING SOLID STATE DISK TO WRITE DATA

TECHNICAL FIELD

[0001] Embodiments of the present invention relate to the field of storage technologies, and in particular, to an array controller, a solid state disk, and a method for controlling a solid state disk to write data.

BACKGROUND

[0002] A Flash Memory (flash memory) apparatus is a non-volatile memory. A storage medium of the flash memory apparatus is a Flash memory chip that is characterized in that no data disappears after power-off. Therefore, the flash memory apparatus is widely used as an external or internal memory. The flash memory apparatus using the Flash memory chip as the storage medium may be a solid state disk (Full name: Solid State Device, SSD for short), which is also known as a solid state drive (Full name: Solid State Drive, SSD for short), or may be another memory.

[0003] One SSD generally includes multiple flash memory chips, where each flash memory chip includes some blocks (block). Generally, when storing data received from the outside, the SSD may concurrently write the data to multiple blocks to improve data processing efficiency. When performing garbage collection processing, the SSD needs to acquire valid data from multiple blocks and move the valid data to a free block, which causes write amplification.

SUMMARY

[0004] Embodiments of the present invention propose an array controller, a solid state disk, and a method for controlling a solid state disk to write data, which can write a segment of data whose size is equal to an integer multiple of a standard size of a block into one or more blocks, so as to fill the one or more blocks, thereby reducing write amplification when the solid state disk performs garbage collection.

[0005] A first aspect of the embodiments provides an array controller, where the array controller is located in a storage system, the storage system further includes a solid state disk, and the array controller includes a communications interface and a processor, where the communications

interface is configured to communicate with the solid state disk; and the processor is configured to receive information about a logical block sent by the solid state disk, where the information about the logical block includes a size of the logical block and indication information of the logical block, and the processor sends multiple write data requests to the solid state disk, where each write data request carries target data, each write data request is used to instruct the solid state disk to write the target data into the logical block indicated by the indication information of the logical block, and a sum of lengths of the target data carried in the multiple write data requests is equal to the size of the logical block. After receiving the multiple write data requests, the solid state disk writes, according to indication information in each write data request, the target data carried in the write data request into the logical block indicated by the indication information. It should be noted that because the logical block includes one or more blocks, writing the target data into the logical block is actually writing the target data into the one or more blocks included in the logical block.

[0006] In this implementation manner, because the sum of the lengths of the target data carried in the multiple write data requests that are sent by the processor to the solid state disk is equal to the size of the logical block, and each write data request is used to instruct the solid state disk to write the target data into the logical block indicated by the indication information of the logical block, the solid state disk can write, according to the indication information, all of the target data carried in the multiple write data requests into the same logical block, and fill the logical block. Because in this embodiment, one logical block includes one or more blocks, it means that the target data is written into the one or more blocks, and these blocks are filled. It may be understood by a person skilled in the art that during garbage collection, the solid state disk performs collection in blocks. Because the target data is all written into the one or more blocks and the one or more blocks do not include other data, if the target data is all released by the array controller subsequently, that is, marked as invalid data, the SSD may directly erase data from all the blocks in the logical block without moving valid data, thereby reducing write amplification during garbage collection.

[0007] With reference to the first aspect, in a first implementation manner of the first aspect, the blocks included in the logical block are located in different channels. Generally, within the solid state disk, concurrent reads and writes may be implemented between channels. Therefore, when the blocks included in the logical block are located in different channels, the target data can be concurrently written into the logical block, which improves data write efficiency.

[0008] With reference to the first aspect, in a second implementation manner of the first aspect, the blocks included in the logical block belong to a same channel. In some cases, concurrent reads and writes may also be implemented between multiple blocks in one channel. Therefore, when the blocks included in the logical block belong to a same channel, the target data can be concurrently

written into the logical block, which can also improve write data efficiency.

[0009] With reference to either of the foregoing implementation manners of the first aspect, in a third implementation manner of the first aspect, the logical block is a logical block in a to-be-written state, where the logical block in the to-be-written state refers to a logical block that has been allocated by the solid state disk to store data. In this embodiment, a logical block has four states: free, to-be-written, full, and bad. The solid state disk reports the logical block in the to-be-written state to the array controller, where data can be directly written into the logical block in the to-be-written state. Therefore, when receiving the write data requests sent by the array controller, the solid state disk may directly write the target data into the logical block in the to-be-written state according to the indication information carried in the write data requests.

[0010] With reference to any one of the foregoing implementation manners of the first aspect, in a fourth implementation manner of the first aspect, for the indication information of the logical block, in a specific implementation manner, the indication information is an identifier that is allocated by the solid state disk to the logical block. When the indication information is the identifier of the logical block, each write data request that is sent by the array controller to the solid state disk carries the identifier. After receiving the write data request, the solid state disk may write, according to the identifier, the target data in each write data request into the logical block corresponding to the identifier, so as to fill the logical block corresponding to the identifier.

[0011] With reference to any one of the foregoing implementation manners of the first aspect, this embodiment further provides an implementation manner alternative to the fourth implementation manner of the first aspect, that is, a fifth implementation manner of the first aspect, and in this implementation manner, the indication information of the logical block includes a logical address range that is allocated by the solid state disk to the logical block. When the indication information is the logical address range, each write data request that is sent by the array controller to the solid state disk carries a sub-range of the logical address range. After receiving the write data request, the solid state disk may determine, according to the sub-range, the logical block corresponding to the target data in each write data request, and write the target data into the logical block, so as to fill the logical block.

[0012] With reference to any one of the foregoing implementation manners of the first aspect, in a sixth implementation manner of the first aspect, the array controller further includes a cache, and the processor is further configured to read multiple pieces of target data from the cache. Because the array controller includes the cache, after receiving external data, the array controller may temporarily store the data in the cache instead of directly sending the data to the solid state disk for storage, and send data to the solid state disk when a size of data in the cache reaches a particular

water mark (for example, the size of the logical block).

[0013] With reference to the sixth implementation manner of the first aspect, in a seventh implementation manner of the first aspect, the storage system further includes a host, and the array controller is located between the host and the solid state disk. In this implementation manner, the array controller may receive data from the host, and write the data into the cache.

[0014] A second aspect of the embodiments provides a solid state disk, where the solid state disk includes a flash memory chip and a solid state disk controller, where the flash memory chip includes multiple channels, and each channel includes multiple blocks; the solid state disk controller is configured to send information about a logical block to an array controller, where the information about the logical block includes a size of the logical block and indication information of the logical block; and in addition, the solid state disk controller is further configured to receive multiple write data requests sent by the array controller, where each write data request carries target data, and each write data request is used to instruct the solid state disk to write the target data into the logical block indicated by the indication information of the logical block. In addition, a sum of lengths of the target data carried in the multiple write data requests is equal to the size of the logical block; and the solid state disk controller then writes the target data carried in each write data request into the logical block indicated by the indication information. It should be noted that because the logical block includes one or more blocks, writing the target data into the logical block is actually writing the target data into the one or more blocks included in the logical block.

[0015] In this implementation manner, the solid state disk reports the information about the logical block to the array controller. The information about the logical block includes the size of the logical block and the indication information of the logical block. Therefore, the array controller may send the write data requests to the solid state disk according to the information about the logical block. Specifically, the sum of the lengths of the target data carried in the multiple write data requests that are sent by the array controller to the solid state disk is equal to the size of the logical block, and each write data request is used to instruct the solid state disk to write the target data into the logical block indicated by the indication information of the logical block. Therefore, the solid state disk may write, according to the indication information, all of the target data carried in the multiple write data requests into the same logical block, and fill the logical block. Because in this embodiment, one logical block includes one or more blocks, it means that the target data is written into the one or more blocks, and these blocks are filled. It may be understood by a person skilled in the art that during garbage collection, the solid state disk performs collection in blocks. Because the target data is all written into the one or more blocks and the one or more blocks do not include other data, if the target data is all released by the array controller subsequently, that is, marked as invalid

data, the SSD may directly erase data from all the blocks in the logical block without moving valid data, thereby reducing write amplification during garbage collection.

[0016] With reference to the second aspect, in a first implementation manner of the second aspect, the blocks included in the logical block are located in different channels. Generally, within the solid state disk, concurrent reads and writes may be implemented between channels. Therefore, when the blocks included in the logical block are located in different channels, the target data can be concurrently written into the logical block, which improves data write efficiency.

[0017] With reference to the second aspect, in a second implementation manner of the second aspect, the blocks included in the logical block belong to a same channel. In some cases, concurrent reads and writes may also be implemented between multiple blocks in one channel. Therefore, when the blocks included in the logical block belong to a same channel, the target data can be concurrently written into the logical block, which can also improve write data efficiency.

[0018] With reference to either of the foregoing implementation manners of the second aspect, in a third implementation manner of the second aspect, the logical block is a logical block in a to-be-written state, where the logical block in the to-be-written state refers to a logical block that has been allocated by the solid state disk to store data. In this embodiment, a logical block has four states: free, to-be-written, full, and bad. The solid state disk reports the logical block in the to-be-written state to the array controller, where data can be directly written into the logical block in the to-be-written state. Therefore, when receiving the write data requests sent by the array controller, the solid state disk may directly write the target data into the logical block in the to-be-written state according to the indication information carried in the write data requests.

[0019] With reference to any one of the foregoing implementation manners of the second aspect, in a fourth implementation manner of the second aspect, for the indication information of the logical block, in a specific implementation manner, the indication information is an identifier that is allocated by the solid state disk to the logical block. When the indication information is the identifier of the logical block, each write data request received by the solid state disk carries the identifier. After receiving the write data request, the solid state disk may write, according to the identifier, the target data in each write data request into the logical block corresponding to the identifier, so as to fill the logical block corresponding to the identifier.

[0020] With reference to any one of the foregoing implementation manners of the second aspect, this embodiment further provides an implementation manner alternative to the fourth implementation manner of the second aspect, that is, a fifth implementation manner of the second aspect, and in this implementation manner, the indication information of the logical block includes a logical address range that is allocated by the solid state disk to the logical block. When the

indication information is the logical address range, each write data request received by the solid state disk carries a sub-range of the logical address range. After receiving the write data request, the solid state disk may determine, according to the sub-range, the logical block corresponding to the target data in each write data request, and write the target data into the logical block, so as to fill the logical block.

[0021] With reference to the third to fifth implementation manners of the second aspect, in a sixth implementation manner of the second aspect, the solid state disk controller is further configured to reclaim the indication information allocated to the logical block. Because the indication information is allocated to only the logical block in the to-be-written state, when the logical block is filled, the indication information of the logical block may be reclaimed for another logical block in the to-be-written state for use.

[0022] With reference to any one of the foregoing implementation manners of the second aspect, in a seventh implementation manner of the second aspect, the solid state disk controller is specifically configured to send an SCSI WRITE command to the array controller, where the command includes a GROUP NUMBER field, and the field is used to carry the indication information of the logical block. In this implementation manner, a private command is provided to report the indication information of the logical block. A conventional SCSI command does not include the GROUP NUMBER field. In this embodiment, the GROUP NUMBER field is specially added to the SCSI WRITE command to report the indication information of the logical block.

[0023] A third aspect of the embodiments of the present invention provides a method for controlling a solid state disk to write data, where the method is applied to the array controller provided in the first aspect.

[0024] A fourth aspect of the embodiments of the present invention provides a method for controlling a solid state disk to write data, where the method is applied to the array controller provided in the second aspect.

[0025] A fifth aspect of the embodiments of the present invention provides an apparatus for controlling a solid state disk to write data, where the apparatus is located in the array controller provided in the first aspect.

[0026] A sixth aspect of the embodiments of the present invention provides an apparatus for controlling a solid state disk to write data, where the apparatus is located in the array controller provided in the second aspect.

[0027] According to the methods for controlling a solid state disk to write data and the apparatuses for controlling a solid state disk to write data provided in the third aspect to the sixth aspect, a segment of data whose size is equal to an integer multiple of a standard size of a block can

be written into one or more blocks, so as to fill the one or more blocks, thereby reducing write amplification when the solid state disk performs garbage collection.

[0028] The embodiments of the present invention provide a computer program product, including a computer readable storage medium that stores program code, where an instruction included in the program code may be executed by the array controller in the first aspect, and is used to perform at least one method in the third aspect.

[0029] The embodiments of the present invention provide a computer program product, including a computer readable storage medium that stores program code, where an instruction included in the program code may be executed by the array controller in the second aspect, and is used to perform at least one method in the fourth aspect.

[0030] The foregoing computer program product provided in the embodiments of the present invention can write a segment of data whose size is equal to an integer multiple of a standard size of a block into one or more blocks, so as to fill the one or more blocks, thereby reducing write amplification when a solid state disk performs garbage collection.

BRIEF DESCRIPTION OF DRAWINGS

[0031] To describe the technical solutions in the embodiments of the present invention more clearly, the following briefly introduces the accompanying drawings required for describing the embodiments or in the prior art. Apparently, the accompanying drawings in the following description show merely some embodiments of the present invention, and a person of ordinary skill in the art may still derive other drawings from these accompanying drawings without creative efforts.

[0032] FIG. 1 is a diagram of an application scenario according to an embodiment of the present invention;

[0033] FIG. 2 is a schematic diagram of a flash memory chip according to an embodiment of the present invention;

[0034] FIG. 3 is a schematic diagram of a block according to an embodiment of the present invention;

[0035] FIG. 4 is a schematic diagram of storing data by a solid state disk in the prior art according to an embodiment of the present invention;

[0036] FIG. 5 is a schematic diagram of a logical block according to an embodiment of the present invention;

[0037] FIG. 6 is a schematic structural diagram of an array controller according to an embodiment of the present invention;

[0038] FIG. 7 is a schematic flowchart of a method for controlling a solid state disk to write data according to an embodiment of the present invention;

[0039] FIG. 8 is a schematic diagram of storing data by a solid state disk according to an embodiment of the present invention;

[0040] FIG. 9 is a schematic flowchart of another method for controlling a solid state disk to write data according to an embodiment of the present invention;

[0041] FIG. 10 is a schematic flowchart of a still another method for controlling a solid state disk to write data according to an embodiment of the present invention;

[0042] FIG. 11 is a schematic structural diagram of an apparatus for controlling a solid state disk to write data according to an embodiment of the present invention; and

[0043] FIG. 12 is a schematic structural diagram of another apparatus for controlling a solid state disk to write data according to an embodiment of the present invention.

DESCRIPTION OF EMBODIMENTS

[0044] Embodiments of the present invention propose an array controller, a solid state disk, and a method for controlling a solid state disk to write data, which can write a segment of data whose size is equal to an integer multiple of a standard size of a block into one or more blocks, so as to fill the one or more blocks, thereby reducing write amplification when the solid state disk performs garbage collection.

[0045] FIG. 1 is a composition diagram of a storage system according to an embodiment of the present invention. The storage system shown in FIG. 1 includes a host 33, an array controller 11, and a solid state disk 22. The host 33 may be a terminal device such as a server or a desktop computer.

[0046] The array controller 11 is located between the host 33 and the solid state disk 22, and may be a computing device, for example, a server or a desktop computer. An operating system and another application program are installed in the array controller 11. The array controller 11 may receive an input/output (I/O) request from the host 33, store data carried in the I/O request, and write data stored in the array controller 11 into the solid state disk 22 when a particular condition is satisfied.

[0047] The solid state disk (Solid State Device, SSD) 22 is a memory using a Flash memory chip as a storage medium, which is also referred to as a solid state drive (Solid State Drive, SSD), or may include another memory. In this embodiment, the solid state disk 22 is described by using the SSD as an example.

[0048] FIG. 1 is only exemplary for description, and constitutes no limitation to a specific

networking manner. For example, a cascading tree network or a ring network may be used, provided that the array controller 11 and the solid state disk 22 can communicate with each other.

[0049] The solid state disk 22 includes an SSD controller 220 and a storage medium 221. The SSD controller 220 is configured to perform an operation such as a write data request or a read data request sent by the array controller 11.

[0050] The storage medium 221 generally includes some flash memory (Flash) chips. Each flash memory chip includes multiple channels, and each channel includes some blocks (block) (as shown in FIG. 2). Generally, the flash memory chip may be divided into some blocks according to a fixed size. Therefore, a block has a standard size. The standard size of the block refers to a size of data that can be stored in a block that is idle and that includes no bad page (bad page). Generally, the standard size of the block may be the Nth power of 2 (M), where N is a positive integer. An erase operation performed by the SSD is performed in blocks. For example, when the SSD needs to perform garbage collection, the SSD may first move valid data from one block to another new block, and then erase all data (including the valid data and invalid data) stored in the original block. In this embodiment of the present invention, valid data in a block refers to data that is stored in the block and that has not been modified, and this part of data is possibly to be read; invalid data in a block refers to data that is stored in the block and that has been modified, and this part of data is impossibly to be read. It may be known by a person skilled in the art that because of an erase feature of a flash memory chip, data stored in a block is not directly modified as data in a common mechanical hard drive is. When data in a block needs to be modified, the SSD controller 220 finds a new block and writes modified data into the new block, and the data in the original block becomes invalid data. The invalid data is erased when the SSD performs garbage collection.

[0051] In addition, as shown in FIG. 3, each block includes some pages (page). The SSD writes data in pages when executing a write data request. For example, the array controller 11 sends a write data request to the SSD controller 220, where the write data request carries logical block addresses (logical block Address, LBA) and data, and the LBAs are addresses that are visible and accessible to the array controller 11. After receiving the write data request, the SSD controller 220 writes the data into pages of one or more blocks according to a set policy. Addresses of the pages into which the data is written are addresses at which the data is actually stored, and are also referred to as physical addresses. The SSD includes a flash translation layer (Flash Translation Layer, FTL), configured to establish and store a correspondence between the LBAs and the physical addresses. When the array controller 11 subsequently sends a read data request to the SSD controller 220, to request to read the data, where the read data request carries the LBAs, the SSD controller 220 may read the data according to the LBAs and the correspondence between the LBAs and the physical

addresses, and send the data to the array controller 11.

[0052] When the array controller 11 reads a segment of data from a cache, and sends the segment of data to the SSD, generally, the segment of data is a segment of data with contiguous logical addresses. However, when receiving the write data request, the SSD controller 220 writes, according to the policy set by the SSD controller 220, the data into the pages (that is, the physical addresses) of the one or more blocks. It may be understood by a person skilled in the art that actually, these physical addresses are not necessarily contiguous. That is, from the perspective of the array controller 11, the segment of data with the contiguous logical addresses is actually written into physical space of the SSD in a scattered manner. The following describes how to write data by an SSD in the prior art with reference to FIG. 4.

[0053] FIG. 4 is a schematic diagram of how to write data by an SSD in the prior art. Generally, because the solid state disk 22 may concurrently write data into channels, after the array controller 11 sends data to the solid state disk 22, the SSD controller 220 generally selects one block from each channel, and concurrently writes the data into the multiple blocks. It may be understood that the solid state disk 22 may also concurrently write the data into multiple blocks in one channel. As shown in FIG. 4, after the data is written into the multiple blocks, each block provides a part of space to store a part of the data. In FIG. 4, a white part of a block represents free space of the block, and a gray part represents used space of the block after a part of the data is written into the block. Therefore, from the perspective of the array controller 11, a segment of data with contiguous logical addresses is stored in multiple blocks in a scattered manner, and data belonging to different logical address segments is written into different blocks. It may be known by a person skilled in the art that a garbage collection operation is regularly performed in the SSD, and erasing is performed in blocks during a garbage collection operation. As the segment of data with the contiguous logical addresses is stored in the multiple blocks in a scattered manner, and the data belonging to the different logical address segments is written into the different blocks, during the garbage collection, valid data needs to be moved from multiple blocks to a free block, which increases write amplification caused by the garbage collection. The write amplification means that data written into a flash memory chip is more than data that is written by a host into an SSD.

[0054] In this embodiment, as shown in FIG. 5, one block may be taken out of each channel, to form a logical block (logical block). That is, the logical block in this embodiment includes multiple blocks, and the blocks are located in different channels. It may be understood that one block may also be taken out of each of a part of the channels, to form a logical block. Likewise, the multiple blocks included in the logical block are separately located in different channels. A size of the logical block is an integer multiple of a standard size of a block, and depends on a quantity of blocks

included in the logical block. In this embodiment, the size of the logical block may be stored in the array controller 11 in advance. After receiving a write data request sent by the host 33, the array controller 11 may temporarily store the write data request in the cache of the array controller 11, and send data to the SSD for storage when the data in the cache reaches the size of the logical block.

[0055] In this embodiment, logical blocks are classified into four types according to states of the logical blocks: free (free) logical block, to-be-written (or writing) logical block, full (full) logical block, and bad (bad) logical block. A logical block in a free state refers to a logical block in which data has been erased; a logical block in a to-be-written state refers to a logical block that has been allocated for data writing; a logical block in a full state refers to a logical block whose space has been filled, and the logical block in the full state may become a logical block in a free state after erasing; a logical block in a bad state refers to a damaged logical block that cannot be used. A logical block for storing data mentioned in this embodiment is actually a logical block in a to-be-written state. Therefore, unless otherwise stated in the following description, a logical block refers to the logical block in the to-be-written state.

[0056] To identify those logical blocks in the to-be-written state, in an optional implementation manner, the SSD allocates an identifier to each logical block. The identifier may be a digit, a letter, or another symbol for uniquely identifying the logical block, or may be any combination of a digit, a letter, or another symbol. For example, an identifier of a logical block is 0, an identifier of another logical block is 1, an identifier of still another logical block is 2, and so on.

[0057] In another optional implementation manner, the SSD allocates a logical address range to each logical block. For example, a logical address range corresponding to a logical block is 0 MB to 1023 MB, a logical address range corresponding to another logical block is 1024 MB to 2047 MB, a logical address range corresponding to still another logical block is 2048 MB to 3071 MB, and so on.

[0058] The SSD controller 220 needs to report the identifier of the logical block or the logical address range of the logical block to the array controller 11, so that when sending data to the SSD for storage, the array controller 11 can add the identifier of the logical block to a write data request and send the write data request to the SSD, to instruct the SSD to write the data into the logical block corresponding to the identifier. Alternatively, a start logical address and a length in a write data request are designated according to the logical address range of the logical block. If there are multiple logical blocks in the to-be-written state, the array controller 11 receives identifiers of the multiple logical blocks or logical address ranges of the multiple logical blocks, then selects any one of the multiple logical blocks, and sends a write data request to the SSD according to an identifier

or a logical address range of the logical block.

[0059] The following describes a hardware structure of the array controller 11. FIG. 6 is a schematic structural diagram of the array controller 11 according to an embodiment of the present invention. As shown in FIG. 6, the array controller 11 mainly includes a processor (processor) 118, a cache (cache) 120, a memory (memory) 122, a communications bus (bus for short) 126, and a communications interface (Communications Interface) 128. The processor 118, the cache 120, the memory 122, and the communications interface 128 communicate with each other by using the communications bus 126.

[0060] The communications interface 128 is configured to communicate with the host 33 or the solid state disk 22.

[0061] The processor 118 may be a central processing unit CPU or an application specific integrated circuit ASIC (Application Specific Integrated Circuit), or is configured as one or more integrated circuits that implement this embodiment of the present invention. In this embodiment of the present invention, the processor 118 may be configured to receive a write data request or a read data request from the host 33, process the write data request or the read data request, send the write data request or the read data request to the solid state disk 22, and perform another operation.

[0062] The memory 122 is configured to store a program 124. The memory 122 may include a high-speed RAM memory, and may further include a non-volatile memory (non-volatile memory), such as at least one magnetic disk memory. It may be understood that the memory 122 may be any non-transitory (non-transitory) computer-readable medium that can store program code, such as a random access memory (Random Access Memory, RAM), a magnetic disk, a hard disk, an optical disc, a solid state disk (Solid State Disk, SSD), or a non-volatile memory.

[0063] The program 124 may include program code, where the program code includes a computer operation instruction.

[0064] The cache 120 (Cache) is configured to temporarily store data received from the host 33 or data read from the solid state disk 22. The cache 120 may be any non-transitory (non-transitory) machine-readable medium that can store data, such as a RAM, a ROM, a flash memory (Flash memory), or a solid state disk (Solid State Disk, SSD), which is not limited herein. For example, when receiving a write data request sent by the host 33, the array controller 11 may store the write data request in the cache 120, and the processor 118 processes the write data request. Optionally, when receiving multiple write data requests sent by the host 33, the array controller 11 may temporarily store the multiple write data requests in the cache 120, and when a size in the cache 120 reaches a water mark (for example, a size of data stored in the cache 120 reaches a size of a logical block), read data whose size is equal to the size of the logical block from the cache 120, and send

the data to the solid state disk 22 for persistent storage.

[0065] In addition, the memory 122 and the cache 120 may be integrated or disposed separately, which is not limited in this embodiment of the present invention.

[0066] The following describes a process of a method for writing data into a solid state disk according to an embodiment of the present invention. The method for writing data into a solid state disk in this embodiment of the present invention may be applied to the storage system shown in FIG. 1 and the array controller 11 shown in FIG. 6. As shown in FIG. 7, the method includes the following steps:

[0067] Step S301: An SSD controller 220 sends indication information of a logical block to an array controller 11.

[0068] Specifically, the SSD controller 220 sends the indication information of the logical block to a processor 118 of the array controller 11. The indication information of the logical block may be an identifier that is allocated by a solid state disk 22 to the logical block, or may be a logical address range that is allocated by a solid state disk 22 to the logical block, or other information for indicating a particular logical block.

[0069] Step S302: The SSD controller 220 sends a size of the logical block to the array controller 11.

[0070] Specifically, the SSD controller 220 sends the size of the logical block to the processor 118 of the array controller 11, and the processor 118 receives the size of the logical block. It should be noted that step S302 and step S301 are not necessarily performed in a particular order. That is, in this embodiment, provided that the SSD controller 220 sends the size of the logical block to the array controller 11, a time period or step in which the size of the logical block is sent to the array controller 11 is not limited. Moreover, the information may be sent separately, or may be sent together with other information, for example, the identifier of the logical block.

[0071] Step S303: The array controller 11 sends multiple pieces of target data to the SSD controller 220, where a sum of lengths of the multiple pieces of target data is equal to the size of the logical block.

[0072] Specifically, the processor 118 of the array controller 11 adds the multiple pieces of target data to multiple write data requests, and sends the multiple write data requests to the SSD controller 220.

[0073] When the indication information is the identifier of the logical block, each write data request further needs to carry the identifier of the logical block. When the indication information is the logical address range of the logical block, each write data request further needs to carry a sub-range of the logical address range. These two cases are described in detail in implementation

manners shown in FIG. 9 and FIG. 10.

[0074] Step S304: The SSD controller 220 writes the multiple pieces of target data into the logical block.

[0075] Specifically, after receiving the multiple write data requests in step S303, the SSD controller 220 may determine, according to indication information in each write data request, the logical block indicated by the indication information, and then write the target data in each write data request into the logical block.

[0076] In the prior art, a write data request that is sent by the array controller 11 to the solid state disk 22 does not include the indication information of the logical block. Therefore, after receiving the write data request, the SSD controller 220 in the prior art generally selects multiple blocks and concurrently writes data into the multiple blocks, which causes the data to be stored in the blocks in a scattered manner. However, in this embodiment, the processor 118 adds the identifier of the logical block to the multiple write data requests and sends the multiple write data requests to the solid state disk 22, to instruct the solid state disk 22 to write all the target data in these write data requests into the logical block indicated by the indication information. It should be noted that because the logical block includes one or more blocks, writing the target data into the logical block is actually writing the target data into the one or more blocks included in the logical block.

[0077] Therefore, in step S304, after receiving the multiple write data requests, the SSD controller 220 may write the target data carried in each write data request into the logical block indicated by the indication information. Moreover, because the sum of the lengths of the target data is equal to the size of the logical block, the logical block is just filled after the SSD controller 220 writes the target data carried in the write data requests into the logical block (as shown in FIG. 8).

[0078] According to the implementation manner shown in FIG. 7, the multiple pieces of target data are stored in one logical block, that is, one or more blocks. Because these pieces of target data are all written into the one or more blocks, and the one or more blocks do not include other data, if these pieces of target data are all released by the array controller subsequently, that is, marked as invalid data, an erase operation may be directly performed on the blocks without moving valid data, which reduces a problem of write amplification during garbage collection.

[0079] Specifically, when the array controller 11 needs to perform a garbage collection operation on the solid state disk, the array controller 11 may generally send a read command to the solid state disk 22, to request to read valid data in a logical block of the solid state disk 22. After the solid state disk 22 sends the valid data to the array controller 11, the array controller 11 sends a write command to the solid state disk 22, to request to write the valid data into a new logical block. Then, the array controller 11 sends a trim command to the solid state disk 22, to indicate that the

original logical block becomes invalid. When performing a garbage collection operation within the solid state disk, the solid state disk finds that the logical block is invalid and may consider that all data stored in the logical block is invalid data, thereby directly performing erasing data from blocks included in the logical block without acquiring the valid data or moving the valid data again. In this way, a problem of write amplification when the solid state disk performs garbage collection is reduced.

[0080] The following describes a process of a method for writing data into a solid state disk according to an embodiment of the present invention. The method for writing data into a flash memory apparatus in this embodiment of the present invention may be applied to the storage system shown in FIG. 1 and the array controller 11 shown in FIG. 6. As shown in FIG. 8, the method includes the following steps:

[0081] Step S101: An SSD controller 220 determines a logical block in a to-be-written state from multiple logical blocks included in an SSD.

[0082] As described above, logical blocks are generally in four states: free, to-be-written, full, and bad. In this embodiment, logical blocks in the to-be-written state are selected from these logical blocks, and are put into a logical block queue. When data needs to be written, any logical block may be directly acquired from the logical block queue for data writing. The logical block queue is a data structure for managing logical blocks in the to-be-written state. It may be understood that in this embodiment, in addition to the queue, another data structure, for example, a linked list, may also be used to manage the logical blocks. The logical block queue may be stored in a cache of the SSD controller 220 or a flash memory chip.

[0083] There may be one or more determined logical blocks in the to-be-written state.

[0084] Step S102: The SSD controller 220 allocates an identifier to the logical block in the to-be-written state.

[0085] The identifier may be a number, for example, 0, 1, 2, …, or may be another symbol for uniquely identifying the logical block in the to-be-written state, or may be any combination of a digit, a letter, or another symbol.

[0086] Step S103: The SSD controller 220 sends the identifier that is allocated to the logical block in step S102 to an array controller 11.

[0087] In this embodiment, the SSD controller 220 may send the identifier to the array controller 11 by using an SCSI WRITE command, where the SCSI WRITE command includes a GROUP NUMBER field, and the field may be used to carry the identifier.

[0088] Step S104: The SSD controller 220 sends a size of the logical block to the array controller 11.

[0089] This step is similar to step S302 in FIG. 7, and details are not described herein again.

[0090] Step S105: The array controller 11 receives multiple pieces of data sent by a host 33.

[0091] The multiple pieces of data may be sent by the host 33 to the array controller 11 by using multiple write commands. Specifically, a processor 118 of the array controller 11 receives the multiple pieces of data sent by the host 33, where each write command carries at least one piece of data and a host logical address range of the data. A host logical address refers to an address of storage space that is presented by the array controller 11 to the host 33. Host logical address ranges of data may be not contiguous. For example, a host logical address range of a piece of data is 0x40000 to 0x40020, a host logical address range of another piece of data is 0x10010 to 0x10020, and the host logical address ranges of these two pieces of data are not contiguous. It should be noted that in actual application, the write command may carry a start host logical address and a length, and the host logical address range is obtained according to the start host logical address and the length.

[0092] Step S106: The array controller 11 writes the received multiple pieces of data into a cache 120.

[0093] Specifically, the processor 118 of the array controller 11 writes the received multiple pieces of data into the cache 120.

[0094] Generally, after receiving the data sent by the host 33, the array controller 11 temporarily writes the data into the cache 120 instead of directly writing the data into a solid state disk 22. When storage space of the cache 120 reaches a particular water mark, the processor 118 reads a part of or all data from the cache 120, and sends the data to the SSD 22 for persistent storage.

[0095] It should be noted that steps S105 and S106 and steps S101 to S104 are not necessarily perform in a particular order. That is, a process of receiving and temporarily storing, by the array controller 11, target data sent by the host 33 and a process of receiving information about the logical block sent by the SSD 22 do not conflict, and are not necessarily executed in a particular order, and may be concurrently executed.

[0096] Step S107: The array controller 11 reads multiple pieces of target data from the cache 120.

[0097] Specifically, the processor 118 of the array controller 11 reads the multiple pieces of target data from the cache 120.

[0098] A sum of lengths of the multiple pieces of target data is equal to the size of the logical block. In this embodiment, to be distinguished from the data that is received by the array controller 11 from the host 33, data read from the cache 120 is referred to as target data. The sum of the lengths of these pieces of target data is equal to the size of the logical block. When the processor

118 generates multiple write data requests and sends the multiple write data requests to the solid state disk 22, it may be considered that each write data request carries one piece of target data. It may be understood that when a size of the data that is received by the array controller 11 from the host 33 and that is stored in the cache 120 exceeds the size of the logical block, the processor 118 needs to read only multiple pieces of target data whose size is equal to the size of the logical block from the cache 120. Moreover, when a size of the data that is received by the processor 118 from the host 33 and that is stored in the cache 120 does not reach the size of the logical block, the processor 118 may not read data from the cache 120, and perform processing until the data in the cache 120 reaches the size of the logical block.

[0099] Step S108: The array controller 11 sends multiple write data requests to a solid state disk 22, where each write data request carries target data, a sum of lengths of the target data carried in the multiple write data requests needs to be equal to the size of the logical block, and each write data request carries the identifier of the logical block.

[0100] Specifically, the processor 118 of the array controller 11 sends the multiple write data requests to the solid state disk 22.

[0101] It may be understood that if the solid state disk 22 sends identifiers of multiple logical blocks to the array controller 11, the array controller 11 may select any identifier from the identifiers, and add the identifier to the multiple write data requests.

[0102] In addition, in this embodiment, the write data request that is sent by the array controller 11 to the solid state disk 22 may also include a start logical address and a length (or a logical address range) of each piece of target data. The logical address herein is different from the host logical address in step S105. The logical address herein is an address of storage space that is presented by the solid state disk 22 to the array controller 11, while the host logical address in step S105 is the logical address of the storage space that is presented by the array controller 11 to the host 33. In addition, it should be noted that when the array controller 11 designates the logical address range, logical address ranges of all the target data are contiguous. The array controller 11 sends the start logical address and the length of each piece of target data to the solid state disk 22. After writing the target data into blocks included in the logical block, the SSD controller 220 may store, in an FTL, a correspondence between the start logical addresses and physical addresses into which the target data is actually written. If the array controller 11 needs to read the target data subsequently, the array controller 11 may add the start logical address and the length of the target data to a read data request, and send the read data request to the solid state disk 22. The SSD controller 220 may acquire a physical address of the target data according to the logical address, the length, and the correspondence stored in the FTL, so as to read the target data and send the target

data to the array controller 11.

[0103] Step S110: The SSD controller 220 reclaims the identifier of the logical block.

[0104] Because the logical block has been filled with data, and is in the full (full) state, the SSD controller 220 may reclaim the identifier, delete the logical block from the logical block queue, and allocate the identifier to another logical block in the to-be-written state.

[0105] The following describes a process of another method for writing data into a solid state disk according to an embodiment of the present invention. The method for writing data into a flash memory apparatus in this embodiment of the present invention may be applied to the storage system shown in FIG. 1 and the array controller 11 shown in FIG. 6. As shown in FIG. 10, the method includes the following steps:

[0106] Step S201: An array controller 11 receives multiple pieces of data sent by a host 33. This step is similar to step S105 shown in FIG. 8, reference may be made to the description of step S105, and details are not described herein again.

[0107] Step S202: The array controller 11 writes the received multiple pieces of data into a cache 120. This step is similar to step S106 shown in FIG. 8, reference may be made to the description of step S106, and details are not described herein again.

[0108] Step S203: The array controller 11 receives a logical address range of a logical block sent by an SSD controller 220.

[0109] Specifically, a processor 118 of the array controller 11 receives the logical address range of the logical block. In this embodiment, each logical block corresponds to one logical address range. For example, a logical address range corresponding to a logical block is 0 MB to 1023 MB, a logical address range corresponding to another logical block is 1024 MB to 2047 MB, and a logical address range corresponding to still another logical block is 2048 MB to 3071 MB.

[0110] These logical blocks are all logical blocks in the to-be-written state, and the SSD controller 220 needs to report the logical address ranges of these logical blocks to the array controller 11.

[0111] Step S204: The array controller 11 receives a size of the logical block sent by the SSD controller 220. This step is similar to step S104 shown in FIG. 8, reference may be made to the description of step S104, and details are not described herein again.

[0112] Likewise, steps S201 to S203 and step S204 are not necessarily performed in a particular order.

[0113] Step S205: The array controller 11 reads target data from the cache 120. This step is similar to step S107 shown in FIG. 8, reference may be made to the description of step S107, and details are not described herein again.

[0114] Step S206: The array controller 11 sends multiple write data requests to a solid state disk 22.

[0115] Specifically, the processor 118 of the array controller 11 sends the multiple write data requests to the solid state disk 22, where each write data request may carry target data, a sum of lengths of the target data carried in the multiple write data requests needs to be equal to the size of the logical block, and each write data request carries a start logical address and a length of the target data. The start logical address of the target data and an end logical address of the target data both belong to a logical address range of a same logical block, and the end logical address is obtained according to the start logical address and the length. Therefore, it may be considered that the logical address range of the target data carried in each write data request is a sub-range of the logical address range of the logical block. When receiving logical address ranges of multiple logical blocks, the array controller 11 may select any one from the logical address ranges to determine the start logical address and the length of the target data.

[0116] Step S207: After receiving the multiple write data requests, the SSD controller 220 determines a logical block that corresponds to target data carried in each write data request.

[0117] In this embodiment, because each logical block corresponds to one logical address range, the SSD controller 220 may determine, according to the start logical address and the length that are carried in each write data request and the logical address range corresponding to each logical block, the logical block corresponding to each piece of target data. For example, for target data carried in one of the write data requests, if a start logical address is 1010 MB and a length is 10 KB, a logical block corresponding to the target data is a logical block whose logical address range is 0 MB to 1023 MB.

[0118] Step S208: The SSD controller 220 writes the target data carried in each write data request into the corresponding logical block.

[0119] According to the implementation manner shown in FIG. 10, the array controller 11 may also write a segment of data with contiguous logical addresses into one or more blocks.

[0120] With reference to the implementation manners shown in FIG. 9 and FIG. 10, in another implementation manner of the present invention, the solid state disk 22 may divide storage space of a flash memory chip 222 into at least two parts, where each part includes some logical blocks. For one of the parts, the solid state disk 22 allocates an identifier to each logical block included in the part; for the other part, the solid state disk 22 allocates a logical address range to each logical block included in the part. The identifier of each logical block in the first part and the logical address range of each logical block in the second part are reported to the array controller 11. After reading multiple pieces of target data whose lengths are equal to a size of a logical block from the cache 120,

the array controller 11 adds the multiple pieces of target data to multiple write data requests, and sends the multiple write data requests to the solid state disk 22, where each write data request of the multiple write data requests may include an identifier of the logical block, or may include a start logical address and a length (which may be considered as a sub-range of a logical address range). That is, in this implementation manner, the array controller 11 may send the multiple write data requests to the solid state disk 22 in the implementation manner shown in FIG. 9, or may send the multiple write data requests to the solid state disk 22 in the implementation manner shown in FIG. 10. For the solid state disk 22, when a write data request received by the solid state disk 22 carries an identifier of a logical block, the solid state disk 22 determines the logical block according to the identifier and writes data into the logical block; when a write data request received by the solid state disk 22 carries a sub-range of a logical address range, the solid state disk 22 may determine a logical block according to the sub-range and the logical address range corresponding to each logical block and write data into the logical block.

[0121] Optionally, in the foregoing implementation manner, the storage space of the flash memory chip 222 may further include another part. In the part, the solid state disk 22 may determine a block into which data sent by the array controller 11 is written. That is, a write data request that is sent by the array controller 11 to the solid state disk 22 may not carry an identifier of a logical block or a logical address range of a logical block. In this case, the solid state disk 22 selects one or more blocks for data writing, instead of writing the data into a logical block designated by the array controller 11.

[0122] It may be understood that any two of the foregoing three types of storage space division and implementation manners may be combined, which is more flexible compared with execution of only the implementation manner shown in FIG. 9 or FIG. 10, and can satisfy multiple requirements.

[0123] As shown in FIG. 11, an embodiment of the present invention further provides an apparatus 700 for controlling a solid state disk to write data, where the solid state disk is located in a storage system, the storage system further includes an array controller, and the apparatus is located in the array controller. Specifically, the apparatus includes a receiving module 701 and a processing module 702, where

the receiving module 701 is configured to receive information about a logical block sent by the solid state disk, where the information about the logical block includes a size of the logical block and indication information of the logical block; and

the processing module 702 is configured to send multiple write data requests to the solid state disk, where each write data request carries target data, each write data request is used to instruct the solid state disk to write the target data into the logical block indicated by the indication

information of the logical block, and a sum of lengths of the target data carried in the multiple write data requests is equal to the size of the logical block.

[0124] Optionally, the solid state disk includes a flash memory chip, where the flash memory chip includes multiple channels, each channel includes multiple blocks, the logical block includes one or more blocks, and the blocks included in the logical block are located in different channels.

[0125] Optionally, the logical block is a logical block in a to-be-written state, where the logical block in the to-be-written state refers to a logical block that has been allocated by the solid state disk to store data.

[0126] Optionally, the indication information of the logical block includes an identifier that is allocated by the solid state disk to the logical block, and each write data request includes the identifier of the logical block.

[0127] Optionally, the indication information of the logical block includes a logical address range that is allocated by the solid state disk to the logical block, and each write data request includes a sub-range of the logical address range.

[0128] As shown in FIG. 12, an embodiment further provides another apparatus 800 for controlling a solid state disk to write data, where the solid state disk includes a flash memory chip and a solid state disk controller, the flash memory chip includes multiple channels, each channel includes multiple blocks, and the apparatus is located in the solid state disk controller. The apparatus specifically includes a transceiver module 801 and a write module 802, where

the transceiver module 801 is configured to: send information about a logical block to an array controller, where the information about the logical block includes a size of the logical block and indication information of the logical block; and receive multiple write data requests sent by the array controller, where each write data request carries target data, each write data request is used to instruct the solid state disk to write the target data into the logical block indicated by the indication information of the logical block, and a sum of lengths of the target data carried in the multiple write data requests is equal to the size of the logical block; and

the write module 802 is configured to write the target data carried in each write data request into the logical block indicated by the indication information.

[0129] Optionally, the logical block includes one or more blocks, and the blocks included in the logical block are located in different channels.

[0130] Optionally, the apparatus 800 further includes an allocating module 803, configured to: determine that the logical block is a logical block in a to-be-written state, where the logical block in the to-be-written state refers to a logical block that has been allocated to store data; and allocate the indication information to the logical block in the to-be-written state.

[0131] Optionally, the indication information of the logical block includes an identifier of the logical block, each write data request includes the identifier of the logical block, and the write module is specifically configured to write, according to the identifier of the logical block, the target data in each write data request into the logical block corresponding to the identifier.

[0132] Optionally, the indication information of the logical block includes a logical address range of the logical block, each write data request includes a sub-range of the logical address range, and the write module is specifically configured to determine, according to the sub-range of the logical address range in each write data request and the logical address range of the logical block, the logical block corresponding to the target data in each write data request, and write the target data in each write data request into the corresponding logical block.

[0133] Optionally, the allocating module 803 is further configured to reclaim the indication information allocated to the logical block.

CLAIMS

1. A storage system, comprises:

a solid state disk comprising a plurality of logical blocks, each of the plurality of logical blocks corresponds to multiple blocks comprised in a memory chip of the solid state disk;

a controller, being configured to send a first write data request to the solid state disk, wherein the write data request comprises a first information of a target logical block;

the solid state disk, being configured to identify the target logical block from the plurality of logical blocks according to the first information of the target logical block and write first target data of the first write data request to the target logical block.

2. The storage system according to claim 1, wherein the solid state disk is further configured to allocate second information to the target logical block of the plurality of logical blocks.

3. The storage system according to claim 2, wherein the solid state disk is further configured to send the second information of the target logical block to the controller.

4. The storage system according to claim 2 or 3, wherein the first information and the second information both are an identifier of the target logical block.

5. The storage system according to claim 2 or 3, wherein the first information of the target logical block comprises a logical address range of the first target data, the second information of the target logical block comprises a logical address range of the target logical block, wherein the logical address range of the first target data is within the logical address range of the target logical block.

6. The storage system according to any one of claims 1 to 5, wherein the solid state disk is further configured to set a to-be-written state or a free state to the target logical block.

7. The storage system according to any one of claims 1 to 6, the controller is further configured to send at least one second write data request to the solid state disk, and

the solid state disk is further configured to write second target data of the at least one second write data request to the target logical block, and a sum of lengths of the first target data and the second target data is equal to the size of the target logical block.

8. The storage system according to claim 7, a state of the target logical block is changed from a to-be-written state to a full state after the target logical block is full.

9. The storage system according to claim 8, wherein a state of the target logical block is changed from the full state to a free state by clearing all the data in the target logical block.

10. The storage system according to any one of claims 1 to 9, wherein each of the plurality of logical blocks has an equivalent or similar size.

11. A data processing method, comprises:

sending, by a controller, a first write data request to a solid state disk, wherein the write data

request comprises a first information of a target logical block, and the solid state disk comprises a plurality of logical blocks, each of the plurality of logical blocks corresponds to multiple blocks comprised in a memory chip of the solid state disk;

identifying, by the solid state disk, the target logical block from the plurality of logical blocks according to the first information of the target logical block and write first target data of the first write data request to the target logical block.

12. The method according to claim 11, further comprising:

allocating, by the solid state disk, second information to the target logical block of the plurality of logical blocks.

13. The method according to claim 12, further comprising:

sending, by the solid state disk, the second information of the target logical block to the controller.

14. The method according to claim 12 or 13, wherein the first information and the second information both are an identifier of the target logical block.

15. The method according to claim 12 or 13, wherein the first information of the target logical block comprises a logical address range of the first target data, the second information of the target logical block comprises a logical address range of the target logical block, wherein the logical address range of the first target data is within the logical address range of the target logical block.

16. The method according to any one of claims 11 to 15, further comprising:

setting, by the solid state disk, a to-be-written state or a free state to the target logical block.

17. The method according to any one of claims 1 to 6, the controller is further configured to send at least one second write data request to the solid state disk, and write second target data of the at least one second write data request to the target logical block, and a sum of lengths of the first target data and the second target data is equal to the size of the target logical block.

18. The method according to claim 17, wherein a state of the target logical block is changed from a to-be-written state to a full state after the target logical block is full.

19. The method according to claim 18, wherein a state of the target logical block is changed from the full state to a free state by clearing the data in the target logical block.

20. The method according to any one of claims 11 to 19, wherein each of the plurality of logical blocks has an equivalent or similar size.

21, A controller, comprising a communications interface and a processor, wherein

the communications interface is configured to communicate with a solid state disk; and

the processor is configured to send a first write data request to a solid state disk, wherein the write data request comprises a first information of a target logical block and is for instructing the solid state disk to write a first target data of the first write data request to the target logical block, and the

target logical block corresponds to multiple blocks comprised in a memory chip of the solid state disk.

22. The controller according to claim 21, wherein the processor is further configured to receive a second information of the target logical block from the solid state disk.

23. The controller according to claim 21 or 22, wherein the first information and the second information both are an identifier of the target logical block.

24. The controller according to claim 21 or 22, wherein the first information of the target logical block comprises a logical address range of the first target data, the second information of the target logical block comprises a logical address range of the target logical block, wherein the logical address range of the first target data is within the logical address range of the target logical block.

25. The controller according to any one of claims 21 to 25, wherein the target logical block comprise a to-be-written state, a free state, and a full state.

26. The controller according to any one of claims 21 to 26, the controller is further configured to send at least one second write data request to the solid state disk, wherein the at least one second write data request is for instructing the solid state disk write second target data of the at least one second write data request to the target logical block, and a sum of lengths of the first target data and the second target data is equal to the size of the target logical block.

27. The controller according to claim 26, a state of the target logical block is changed from a to-be-written state to a full state after the target logical block is full.

28. The controller according to claim 27, wherein a state of the target logical block is changed from the full state to a free state by clearing the data in the target logical block.

29. A solid state disk, comprises:

a flash memory chip, comprising multiple blocks; and

a processor, being configured to:

form a plurality of logical blocks, each of the plurality of logical blocks comprising at least two blocks of flash memory chip;

receive a first write data request from a controller, wherein the write data request comprises a first information of a target logical block;

identify the target logical block from the plurality of logical blocks according to the first information of the target logical block; and

write first target data of the first write data request to the target logical block.

30. The solid state disk according to claim 29, wherein the solid state disk controller is further configured to allocate second information to the target logical block of the plurality of logical blocks.

31. The solid state disk according to claim 30, wherein the processor is further configured to send the second information of the target logical block to the controller.

32. The solid state disk according to claim 30 or 31, wherein the first information and the second information both are an identifier of the target logical block.

33. The solid state disk according to claim 30 or 31, wherein the first information of the target logical block comprises a logical address range of the first target data, the second information of the target logical block comprises a logical address range of the target logical block, wherein the logical address range of the first target data is within the logical address range of the target logical block.

34. The solid state disk according to any one of claims 29 to 33, wherein the processor is further configured to set a to-be-written state or a free state to the target logical block.

35. The solid state disk according to any one of claims 29 to 34, the processor is further configured to receive at least one second write data request from the controller, and write second target data of the at least one second write data request to the target logical block, and a sum of lengths of the first target data and the second target data is equal to the size of the target logical block.

36. The solid state disk according to claim 35, a state of the target logical block is changed from a to-be-written state to a full state after the target logical block is full.

37. The solid state disk according to claim 36, wherein a state of the target logical block is changed from the full state to a free state by clearing the data in the target logical block.

38. The solid state disk according to any one of claims 29 to 37, wherein each of the plurality of logical blocks has an equivalent or similar size.