

DATA ACCESS METHOD AND COMPUTING DEVICE

[0001] This application claims priority to Chinese Patent Application No. 202111441693.4, filed with the China National Intellectual Property Administration on November 30, 2021 and entitled "DATA ACCESS METHOD AND COMPUTING DEVICE", which is incorporated herein
5 by reference in its entirety.

TECHNICAL FIELD

[0002] This application relates to the field of data storage, and in particular, to a data access method and a computing device.

BACKGROUND

[0003] With continuous innovation, improvement, and accelerated evolution of data storage technologies, an implementation mechanism of data storage becomes increasingly complex. Data storage is usually executed by a host-side central processing unit (CPU) accessing a host-side storage device. An execution process continuously occupies and consumes processor resources. It may be understood that, working time of the processor purchased at high costs achieves a better
10 gain when being used to undertake a high-value computing task, and it is clear that using the processor to undertake a data storage implementation work is not a preferential solution.

[0004] Based on this, a conventional technology proposes to add a data processing unit (DPU) connected to a host-side processor, to access the host-side storage device to perform a data storage work originally completed by the host-side processor. In this way, the processor may invest as
20 much computing power as possible in a computing service such as a virtual machine running on the processor. However, the data processing unit is usually disposed outside a host, and hardware needs to be improved, so that the data processing unit can exchange data with the host-side storage device. The conventional technology proposes to connect a host-side hardware storage device to the data processing unit in a hardware cable connection manner, so that the data processing unit
25 may access the hardware storage device of the host. However, setting a hardware cable increases hardware design complexity of the data processing unit.

SUMMARY

[0005] In view of this, a data access method and a computing device are provided. The data access method in embodiments of this application is performed by a data processing unit of the computing device. According to the data access method in embodiments of this application, a hardware connection manner is simplified and hardware design complexity is reduced while it is ensured that the data processing unit can directly access a host-side hardware storage device.

[0006] According to a first aspect, an embodiment of this application provides a data access method, where the method is performed by a data processing unit connected to a processor, and the method includes: writing an access request into a first cache queue of the processor, where the access request includes an identifier of a storage device to be accessed, and the storage device is connected to the processor and not directly connected to the data processing unit; and sending, in a peripheral component interconnect express (PCIe) point-to-point transmission mode, a first instruction to the storage device to which the identifier of the storage device points, where the first instruction includes location information of the access request in the first cache queue, and the first instruction instructs the storage device to obtain the access request from the first cache queue and execute the access request.

[0007] According to the data access method in this embodiment of this application, the data processing unit is first used to write, into the first cache queue of the processor, the access request including the identifier of the storage device to be accessed, so that the access request may be stored in the first cache queue of the processor. Then the first instruction is sent, in the PCIe point-to-point transmission mode, to the storage device to which the identifier of the storage device points, so that the storage device to be accessed can receive the first instruction. The first instruction includes the location information of the access request in the first cache queue, and the first instruction instructs the storage device to obtain the access request from the first cache queue and execute the access request, so that after receiving the first instruction, the storage device can obtain the access request from a corresponding location in the first cache queue and execute the access request according to an instruction of the first instruction. In this manner, when the storage device is connected to the processor and not directly connected to the data processing unit, the data processing unit does not need to access the storage device via the processor of a host, and can directly access the storage device connected to the processor, so that resources of the host are not occupied. In addition, this access manner can simplify a hardware connection manner and reduce hardware design complexity.

[0008] According to the first aspect, in a first possible implementation of the data access method, the access request further includes a type of the access request and an access memory

address, the access memory address is an address allocated by the processor to a memory of the data processing unit, the type of the access request indicates that the access request is a read request or a write request, and the access memory address is an address accessible to the storage device.

[0009] In this manner, when obtaining the access request from the first cache queue, the storage device may determine the type of the access request and address information of an access object, so that whether to perform a read operation or a write operation can be determined based on the type of the access request, and when executing the access request, data access by the data processing unit to the storage device of a corresponding operation type is completed based on the address information of the access object.

[0010] According to the first aspect or the first possible implementation of the first aspect, in a second possible implementation of the data access method, the method further includes: obtaining an execution result of the access request from a second cache queue of the processor.

[0011] Because the data processing unit is connected to the processor, the data processing unit can obtain the execution result of the access request from the second cache queue of the processor, to determine an execution status of the access request. The data processing unit does not need to cache the execution result of the access request, so that occupation of a storage space of the data processing unit can be reduced.

[0012] According to the first aspect or the first possible implementation of the first aspect, in a third possible implementation of the data access method, the method further includes: receiving an execution result that is of the access request and that is transmitted by the storage device in the PCIe point-to-point transmission mode.

[0013] In this manner, the execution result of the access request does not need to be transmitted by the storage device to the data processing unit via the processor, but is directly transmitted by the storage device to the data processing unit. This can improve transmission efficiency of the execution result of the access request.

[0014] According to any one of the first to third possible implementations of the first aspect, in a fourth possible implementation of the data access method, the method further includes: transmitting the address of the memory of the data processing unit to the processor; and receiving the access memory address from the processor.

[0015] In this manner, the data processing unit may obtain the access memory address accessible to the storage device, so that the access memory address can be used as an address for the data processing unit to transmit data in the PCIe point-to-point transmission mode and it is possible that the data processing unit and the storage device transmit data when the data processing unit and the storage device are not directly connected to each other.

[0016] According to any one of the first aspect and the foregoing possible implementations of

the first aspect, in a fifth possible implementation of the data access method, the method further includes: receiving addresses of the first cache queue and the second cache queue that are sent by the processor, where the address of the first cache queue is used to write the access request into the first cache queue of the processor, and the address of the second cache queue is used to obtain the execution result of the access request from the second cache queue.

[0017] In this manner, the data processing unit can find the first cache queue based on the received address of the first cache queue, and when writing the access request into the first cache queue, the data processing unit may directly write the access request. The data processing unit can find the second cache queue based on the received address of the second cache queue, and when obtaining the execution result of the access request from the second cache queue, the data processing unit can directly obtain the execution result of the access request. Therefore, real-time performance and accuracy of writing the access request and obtaining an access request result are ensured.

[0018] According to a second aspect, an embodiment of this application provides a data access method, where the method is applied to a computing device, and the computing device includes a processor, a storage device connected to the processor, and a data processing unit connected to the processor and not directly connected to the storage device. The method includes: The data processing unit writes an access request into a first cache queue of the processor, where the access request includes an identifier of the storage device to be accessed. The data processing unit sends, in a peripheral component interconnect express (PCIe) point-to-point transmission mode, a first instruction to the storage device to which the identifier of the storage device points, where the first instruction instructs a location of the access request in the first cache queue. The storage device obtains the access request from the first cache queue and executes the access request according to the first instruction.

[0019] According to the data access method in this embodiment of this application, the data processing unit is first used to write, into the first cache queue of the processor, the access request including the identifier of the storage device to be accessed, so that the access request may be stored in the first cache queue of the processor. Then the data processing unit is used to send, in the PCIe point-to-point transmission mode, the first instruction to the storage device to which the identifier of the storage device points, so that the storage device to be accessed can receive the first instruction. The first instruction includes location information of the access request in the first cache queue, and the first instruction instructs the storage device to obtain the access request from the first cache queue and execute the access request, so that after receiving the first instruction, the storage device can obtain the access request from a corresponding location in the first cache queue and execute the access request according to an instruction of the first instruction. In this manner,

in the computing device, when the storage device is connected to the processor and not directly connected to the data processing unit, the computing device does not need to access the storage device via the processor of a host, and can directly access the storage device connected to the processor, so that resources of the host are not occupied. In addition, this access manner can simplify a hardware connection manner and reduce hardware design complexity.

[0020] According to the second aspect, in a first possible implementation of the data access method, the access request further includes an access memory address, the access memory address is an address allocated by the processor to a memory of the data processing unit, and the access memory address is an address accessible to the storage device. Executing the access request includes: The storage device writes data into the memory of the data processing unit or reads data from the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode.

[0021] In this manner, the storage device can find an access object, that is, the memory of the data processing unit, based on the access memory address, so that the access request can be executed to write data into the memory of the data processing unit or read data from the memory of the data processing unit, to complete access by the memory of the data processing unit to the storage device.

[0022] According to the first possible implementation of the second aspect, in a second possible implementation of the data access method, the access request further includes a type of the access request, and the type of the access request indicates that the access request is a read request or a write request. When the access request is the read request, executing the access request includes: reading data from the storage device based on the access request, and writing, into the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode, the data read from the storage device. When the access request is the write request, executing the access request includes: obtaining, from the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode, data to be written into the storage device, and writing the obtained data into the storage device.

[0023] When the access request is the read request, the storage device can write data into the memory of the data processing unit, which is equivalent to reading the storage device by the memory of the data processing unit. When the access request is the write request, the storage device can read data from the memory of the data processing unit, which is equivalent to writing the storage device by the memory of the data processing unit. In this manner, corresponding to different types of the access request, the storage device can respond to different operations that meet a requirement of the access request.

[0024] According to the first or second possible implementation of the second aspect, in a third

possible implementation of the data access method, the method further includes: The data processing unit sends the address of the memory of the data processing unit to the processor. The processor allocates the access memory address to the memory of the data processing unit based on the address of the memory of the data processing unit. The processor sends the access memory address to the data processing unit.

[0025] In this manner, the processor may allocate the access memory address to the memory of the data processing unit, so that the access memory address can be used as an address for the data processing unit to transmit data in the PCIe point-to-point transmission mode and it is possible that the data processing unit and the storage device transmit data when the data processing unit and the storage device are not directly connected to each other. Allocating the access memory address to the memory of the data processing unit and sending the access memory address may be performed in a configuration phase, so that the access memory address is directly used after software in the data processing unit generates the access request. This can improve efficiency of executing the access request by the computing device.

[0026] According to any one of the second aspect and the foregoing possible implementations of the second aspect, in a fourth possible implementation of the data access method, the method further includes: The storage device writes an execution result of the access request into a second cache queue of the processor. The data processing unit obtains the execution result of the access request from the second cache queue of the processor.

[0027] According to the second aspect and any one of the first to the third possible implementations of the second aspect, in a fifth possible implementation of the data access method, the method further includes: The data processing unit receives an execution result that is of the access request and that is transmitted by the storage device in the PCIe point-to-point transmission mode.

[0028] According to any one of the second aspect and the foregoing possible implementations of the second aspect, in a sixth possible implementation of the data access method, the method further includes: The data processing unit receives addresses of the first cache queue and the second cache queue that are sent by the processor, where the address of the first cache queue is used to write the access request into the first cache queue of the processor, and the address of the second cache queue is used to obtain the execution result of the access request from the second cache queue.

[0029] In this manner, the data processing unit can find the first cache queue based on the received address of the first cache queue, so that the data processing unit can accurately and quickly perform operations of writing the access request into the first cache queue and obtaining the execution result of the access request from the second cache queue. Creation of the first cache

queue and the second cache queue and sending of the addresses of the first cache queue and the second cache queue may be performed in the configuration phase, so that the addresses of the first cache queue and the second cache queue are directly used after the software in the data processing unit generates the access request. This can improve the efficiency of executing the access request by the computing device.

[0030] According to a third aspect, an embodiment of this application provides a computing device, including a processor, a storage device connected to the processor, and a data processing unit connected to the processor and not directly connected to the storage device. The data processing unit is configured to perform the data access method in one or more of the first aspect or the possible implementations of the first aspect.

[0031] According to a fourth aspect, an embodiment of this application provides a non-volatile computer-readable storage medium, where the non-volatile computer-readable storage medium stores computer program instructions, and when the computer program instructions are executed by a processor, the data access method in one or more of the first aspect or the possible implementations of the first aspect is implemented.

[0032] According to a fifth aspect, an embodiment of this application provides a computer program product including computer-readable code or a non-volatile computer-readable storage medium carrying the computer-readable code. When the computer-readable code is run in an electronic device, a processor in the electronic device performs the data access method in one or more of the first aspect or the possible implementations of the first aspect.

[0033] These aspects and other aspects of this application are more concise and more comprehensive in descriptions of the following (a plurality of) embodiments.

BRIEF DESCRIPTION OF DRAWINGS

[0034] Accompanying drawings included in this specification and constituting a part of this specification and this specification jointly show example embodiments, features, and aspects of this application, and are intended to explain principles of this application.

[0035] FIG. 1 shows an implementation of a data storage function of a computing device in a conventional technology;

[0036] FIG. 2 shows another implementation of a data storage function of a computing device in a conventional technology;

[0037] FIG. 3 shows an example application scenario of a computing device according to an embodiment of this application;

[0038] FIG. 4 shows an example working procedure of a computing device according to an

embodiment of this application;

[0039] FIG. 5 shows an example in which a data processing unit sends an address of a memory of the data processing unit to a processor according to an embodiment of this application;

5 **[0040]** FIG. 6 shows an example in which a processor obtains an access memory address and sends the access memory address according to an embodiment of this application;

[0041] FIG. 7 shows an example in which a processor creates a cache queue and sends an address of the cache queue according to an embodiment of this application;

[0042] FIG. 8 shows an example in which a data processing unit writes an access request into a first cache queue of a processor according to an embodiment of this application;

10 **[0043]** FIG. 9 shows an example in which a data processing unit generates a first instruction and sends the first instruction to a storage device according to an embodiment of this application;

[0044] FIG. 10 shows an example in which a storage device obtains an access request and executes the access request according to an embodiment of this application;

15 **[0045]** FIG. 11 shows an example in which a storage device writes an execution result of an access request into a second cache queue according to an embodiment of this application;

[0046] FIG. 12 shows an example in which a data processing unit obtains an execution result of an access request according to an embodiment of this application;

[0047] FIG. 13 shows another example working procedure of a computing device according to an embodiment of this application;

20 **[0048]** FIG. 14 shows an example in which a storage device writes an execution result of an access request into a data processing unit according to an embodiment of this application;

[0049] FIG. 15 shows a schematic diagram of an example working procedure of a data access method according to an embodiment of this application;

25 **[0050]** FIG. 16 shows a schematic diagram of an example working procedure of a data access method according to an embodiment of this application; and

[0051] FIG. 17 shows a schematic structural diagram of a computing device according to an embodiment of this application.

DESCRIPTION OF EMBODIMENTS

30 **[0052]** The following describes various example embodiments, features, and aspects of this application in detail with reference to the accompanying drawings. Identical reference numerals in the accompanying drawings indicate components that have same or similar functions. Although various aspects of embodiments are illustrated in the accompanying drawings, the accompanying drawings are not necessarily drawn in proportion unless otherwise specified.

[0053] The specific term "example" herein means "used as an example, embodiment, or illustration". Any embodiment described as "example" is not necessarily explained as being superior or better than other embodiments.

[0054] In addition, numerous specific details are given in the following specific embodiments to better describe this application. A person skilled in the art should understand that this application may also be implemented without the specific details. In some embodiments, methods, means, components, and circuits well known by a person skilled in the art are not described in detail, so that a main purpose of this application is highlighted.

[0055] FIG. 1 shows an implementation of a data storage function of a computing device in a conventional technology. A computing device in a hyper-converged infrastructure is used as an example of the computing device shown in FIG. 1. The hyper-converged infrastructure (hyper-converged infrastructure, HCI) is a technology that integrates virtualization computing and storage into a same system platform, and may integrate storage and computing functions into a single computing device (or a computing device cluster, where each computing device provides the computing and storage functions).

[0056] As shown in FIG. 1, for a specific computing device in the hyper-converged infrastructure, a host side of the computing device includes a processor (for example, a central processing unit), a SAS expander, a SAS disk array chip, and a storage device. The storage device may include, for example, a serial attached SCSI hard disk drive/solid-state drive (serial attached SCSI hard disk drive/ solid-state drive, SAS HDD/SSD) and/or a non-volatile memory express solid-state drive (non-volatile memory express solid-state drive, NVMe SSD). The processor and the storage device NVMe SSD are connected through a peripheral component interconnect express (peripheral component interconnect express, PCIe) cable, the processor is further connected to the SAS disk array chip through a PCIe cable, and the SAS disk array chip is further connected to the storage device HDD/SSD via the SAS expander.

[0057] The processor runs a storage virtual machine and a plurality of user virtual machines. The storage virtual machine further runs distributed storage software, and each user virtual machine further runs respective application software. The distributed storage software is used to send an access request to the host-side storage device. The processor may directly access the host-side storage device in a PCIe mode based on the access request, to provide a storage service for the application software in the user virtual machine.

[0058] A disadvantage of this solution is that completing a data storage function by the processor by accessing the storage device occupies a large quantity of processor resources, and consequently, a capability of the processor to support the user virtual machine is reduced.

[0059] To release processing resources and memory of the host-side processor, improve

density of the user virtual machine, and reduce system authorization costs, the conventional technology proposes an improved technical solution based on the solution in FIG. 1. FIG. 2 shows another implementation of a data storage function of a computing device in the conventional technology.

5 **[0060]** As shown in FIG. 2, a data processing unit is connected to a host-side processor (for example, a central processing unit) through a PCIe cable, and the data processing unit is used to replace the host-side processor to run distributed storage software, to provide a data storage service for application software in a user virtual machine in a host. In this way, as many processor computing capabilities as possible may be used to run the application software in the user virtual machine. In this case, a problem of how the data processing unit accesses a host-side storage device needs to be resolved.

[0061] The conventional technology proposes to connect the data processing unit to the host-side storage device in a hardware cable connection manner, so that the data processing unit can access the host-side storage device. The data processing unit may be connected to a storage device 15 NVMe SSD through a PCIe cable, and the data processing unit may be connected to a storage device HDD/SSD through a SAS cable. However, this solution has the following requirements on a hardware facility of the data processing unit.

[0062] First, a SAS disk array chip is usually designed to be inserted into the processor. Therefore, the data processing unit needs to integrate a new SAS disk array chip to connect to a 20 SAS expander. Second, the data processing unit is required to have plenty of PCIe ports, or needs to integrate a PCIe switch chip (which is not shown in the figure), to connect a plurality of memories NVMe SSDs via the PCIe ports or the PCIe switch chip.

[0063] Adding a cable and/or a chip brings problems such as extra power consumption, heat dissipation, space layout, and hardware costs to the data processing unit, and greatly increases 25 complexity of a hardware connection manner and hardware design of the data processing unit.

[0064] In view of this, a data access method and a computing device are proposed. The data access method in embodiments of this application is performed by a data processing unit of a computing device. According to the data access method in embodiments of this application, the hardware connection manner is simplified and the hardware design complexity is reduced while it 30 is ensured that the data processing unit can directly access a host-side hardware storage device.

[0065] FIG. 3 shows an example application scenario of a computing device according to an embodiment of this application.

[0066] As shown in FIG. 3, a host side of the computing device in this embodiment of this application includes a processor (for example, a central processing unit) and a plurality of 35 memories (which may include a storage device NVMe SSD and a storage device HDD/SSD). A

data processing unit is connected to the processor through a PCIe cable, and is not connected to each storage device. The data processing unit is not disposed on the host side of the computing device.

[0067] In a host, the processor may run at least one user virtual machine, and each user virtual machine further runs respective application software. The application software has a data storage requirement. The processor is connected to the storage device NVMe SSD through a PCIe cable, the processor is further connected to a SAS disk array chip through a PCIe cable, and the SAS disk array chip is further connected to the storage device HDD/SSD via a SAS expander.

[0068] The data processing unit may run distributed storage software. When the distributed storage software generates an access request, or when other software (which is not shown in the figure) run by the data processing unit generates the access request, the data processing unit performs the data access method in embodiments of this application, and accesses the host-side storage device in a PCIe point-to-point transmission mode, to provide a data storage service for the application software in the user virtual machine of the host-side processor.

[0069] The computing device in FIG. 3 may be, for example, applied to a hyper-converged infrastructure, or may be applied to an application scenario in which another data processing unit accesses the host-side storage device when the data processing unit is connected to the host-side processor and not connected to the host-side storage device. A specific application scenario of the computing device is not limited in this embodiment of this application.

[0070] FIG. 4 shows an example working procedure of a computing device according to an embodiment of this application. With reference to FIG. 4, the following describes an example method for implementing access by a data processing unit to a host-side hardware storage device according to an embodiment of this application.

[0071] The computing device in this embodiment of this application may be first configured before being used by a user, so that in the computing device, the data processing unit may directly respond to an access request, and no apparatus or module in the computing device needs to be configured when the access request is generated to prepare for responding to the access request. To improve user experience as much as possible, the configuration may be completed, for example, before the computing device is delivered from a factory. When using the computing device, the user may perform various operations on application software run by a processor. Some operations may relate to reading and writing of data. In this case, the application software may generate a corresponding storage requirement. Based on the storage requirement of the application software, or based on the operations related to the reading and writing of data, the processor may indicate distributed storage software or other software (which is not shown in the figure) run by the data processing unit connected to the processor to generate the access request. In this case, the data

processing unit may write the access request into a cache queue of the processor, so that the storage device can obtain the access request from the cache queue and execute the access request, to access the host-side hardware storage device.

[0072] As shown in FIG. 4, in the computing device according to this embodiment of this application, before the distributed storage software generates the access request, the computing device may perform steps S1 to S3 to configure the data processing unit, the processor, and the storage device. After the data processing unit, the processor, and the storage device are configured, when the distributed storage software generates the access request, the computing device may perform steps S4 to S8, so that the data processing unit accesses the host-side hardware storage device.

[0073] With reference to FIG. 4, the following first describes an example method for configuring the data processing unit, the processor, and the storage device.

[0074] Step S1: The data processing unit sends an address of a memory of the data processing unit to the processor. FIG. 5 shows an example in which the data processing unit sends the address of the memory of the data processing unit to the processor according to an embodiment of this application.

[0075] For example, as shown in FIG. 5, the data processing unit includes the memory, used to store data that is used or generated when the data processing unit executes a task or instructions. The memory of the data processing unit may include a data page region, used to store data that may be written by the data processing unit into another apparatus or module (for example, the storage device) in the computing device, and data that is written by the another apparatus or module (for example, the storage device) in the computing device. The address of the memory of the data processing unit may be an address of the data page region. The storage device is disposed on a host side. Therefore, for the storage device, an object to which the storage device can transmit data in a PCIe point-to-point transmission mode should also have an address on the host side. However, the data processing unit is not disposed on the host side. Therefore, the address of the memory of the data processing unit is not an address on the host side, and cannot be used as an address of the object to which the storage device transmits data in the PCIe point-to-point transmission mode. Because the data processing unit is connected to the host-side processor, in step S1, the data processing unit may first send the address of the memory of the data processing unit to the host-side processor. The processor first performs processing to obtain an address (for example, refer to step S2 below) that is on the host side and that corresponds to the address of the memory of the data processing unit.

[0076] Step S2: The processor allocates an access memory address to the memory of the data processing unit based on the address of the memory of the data processing unit, and sends the

access memory address to the data processing unit. FIG. 6 shows an example in which the processor obtains the access memory address and sends the access memory address according to an embodiment of this application.

[0077] For example, as shown in FIG. 6, the processor is disposed on the host side of the computing device. Therefore, an address generated by the processor is an address on the host side, and may be used as the address of the transmission object to which the storage device transmits data in the PCIe point-to-point transmission mode. Based on this, after receiving the address of the memory of the data processing unit, the processor may allocate, in a storage space of the processor, a corresponding storage region to the memory of the data processing unit, where the access memory address may be an address of the storage region. When the access memory address is used as the address of the object to which the storage device transmits data in the PCIe point-to-point transmission mode, the memory that is of the data processing unit and that corresponds to the storage region to which the access memory address belongs may be used as the transmission object to which the storage device transmits data in the PCIe point-to-point transmission mode, to establish a transmission channel in the PCIe point-to-point transmission mode between the storage device and the data processing unit. The access memory address may be used when data is transmitted between the data processing unit and the storage device in the PCIe point-to-point transmission mode (for example, refer to step S6 below).

[0078] Step S3: The processor creates a first cache queue and a second cache queue that correspond to the storage device, and sends storage addresses of the first cache queue and the second cache queue in the processor to the data processing unit and the storage device respectively. FIG. 7 shows an example in which the processor creates a cache queue and sends an address of the cache queue according to an embodiment of this application.

[0079] For example, as shown in FIG. 7, the first cache queue may be a submission queue (submission queue, SQ) used to store the access request, and the second cache queue may be a completion queue (completion queue, CQ) used to store an execution result of the access request. The first cache queue and the second cache queue may be stored in a memory of the processor. The processor is connected to both the data processing unit and the storage device. Therefore, the storage addresses of the first cache queue and the second cache queue of the processor may be transmitted to the data processing unit and the storage device respectively. The data processing unit may write the access request into the first cache queue based on the storage address of the first cache queue in the processor (for example, refer to step S4 below), and may read the execution result of the access request from the second cache queue based on the storage address of the second cache queue in the processor (for example, refer to step S8 below). The storage device may read the access request from the first cache queue based on the storage address of the first cache queue

in the processor (for example, refer to step S6 below), and may write the execution result of the access request into the second cache queue based on the storage address of the second cache queue in the processor (for example, refer to step S7 below). In this manner, the access request and the execution result of the access request may be transmitted between the data processing unit and the storage device.

[0080] Step S3 may be performed before step S1 or step S2, or may be performed simultaneously with step S1 or step S2. An execution sequence of step S3 and step S1 and an execution sequence of step S3 and step S2 are not limited in this application.

[0081] After steps S1 to S3 are performed, the data processing unit, the processor, and the storage device have been configured. With reference to FIG. 4, the following describes an example method for implementing, by the computing device, that the data processing unit accesses the host-side hardware storage device after the data processing unit, the processor, and the storage device are configured.

[0082] Step S4: The data processing unit writes the access request into the first cache queue of the processor.

[0083] For example, in step S3, the data processing unit receives the storage address of the first cache queue. Therefore, a location of the first cache queue in the processor is known to the data processing unit. In this case, in the data processing unit, when the distributed storage software or the another software generates the access request, the data processing unit may write the access request into the first cache queue based on the address of the first cache queue.

[0084] FIG. 8 shows an example in which the data processing unit writes the access request into the first cache queue of the processor according to an embodiment of this application. As shown in FIG. 8, the data processing unit may first convert the access request into a form of a submission queue (a first cache queue) entry (submission queue entry, SQE). The entry obtained through conversion may include operation code (operation code), a scatter gather list (scatter gather list, SGL), an identifier, and the like. The first cache queue may include a plurality of entries. When the access request is written into the first cache queue of the processor, the entry obtained through conversion based on the access request may be written into a location of a specific entry (for example, an entry at a tail of a queue) in the first cache queue of the processor in a direct memory access (direct memory access, DMA) manner.

[0085] The operation code may indicate a type of the access request, and the access request may be classified into different types. For example, the access request may be a request of reading the storage device by the data processing unit (which is equivalent to transmitting data from the storage device to the data processing unit). In this case, the access request may be a read request, and the storage device performs a write operation when executing the access request, that is, the

storage device writes data into the data processing unit (for example, refer to step S6 below). Alternatively, the access request may be a request of writing the storage device by the data processing unit (which is equivalent to transmitting data from the data processing unit to the storage device). In this case, the access request may be a write request, and the storage device performs a read operation when executing the access request, that is, the storage device reads data from the memory of the data processing unit (for example, refer to step S6 below). The operation code indicates the type of the access request, so that the type of the access request can be written into the first cache queue of the processor, and the storage device can obtain the type of the access request from the first cache queue, and determine whether execution of the access request is the read operation or the write operation.

[0086] The scatter gather list may store the access memory address. It can be learned from the related descriptions of step S2 that the access memory address may be used as the address of the transmission object to which the storage device transmits data in the PCIe point-to-point transmission mode. The access memory address is stored via the scatter gather list, so that the access memory address can be written into the first cache queue of the processor, and the storage device can find an object (in this example, the object is the data page region of the data processing unit) of the read operation or the write operation based on the access memory address when executing the access request.

[0087] The identifier may identify a storage device to be accessed. The storage device to be accessed may be any storage device of the host-side storage device. Each host-side storage device may correspond to one first cache queue. In this case, each first cache queue may also include an identifier of a storage device corresponding to the first cache queue. When the access request is written into the first cache queue of the processor, a first cache queue with the same identifier may be found from a plurality of first cache queues based on the identifier included in the access request, and then the access request is written into the first cache queue.

[0088] A person skilled in the art should understand that the entry obtained by converting the access request may include more content, for example, a length of the access request. Specific information included in the access request is not limited in this application.

[0089] Step S5: The data processing unit generates a first instruction including location information of the access request in the first cache queue, and sends, in a PCIe point-to-point transmission mode, the first instruction to the storage device to which the identifier of the access request points, where the first instruction instructs the storage device to obtain the access request from the first cache queue and execute the access request.

[0090] The peripheral component interconnect express point-to-point transmission mode (that is, the PCIe point-to-point transmission mode) is an existing data transmission mode in the

conventional technology, and may enable two devices to perform data transmission even when two devices are not directly connected. In this embodiment of this application, the PCIe point-to-point transmission mode is used, so that the first instruction can be transmitted between the data processing unit and the storage device to be accessed that are not directly connected.

5 **[0091]** The storage device to be accessed that is used as a sending object of the first instruction may be the storage device corresponding to the identifier included in the access request in step S4. The first instruction may include the location information of the access request in the first cache queue, and the first instruction may instruct the storage device to obtain the access request from one first cache queue corresponding to the storage device and execute the access request. In other
10 words, a specific queue from which the storage device obtains the access request is determined based on a correspondence between the first cache queue and the storage device. In this case, after receiving the first instruction, the storage device to be accessed may obtain the access request from a corresponding location in the one first cache queue corresponding to the storage device and then execute the access request according to an instruction of the first instruction.

15 **[0092]** FIG. 9 shows an example in which the data processing unit generates the first instruction and sends the first instruction to the storage device according to an embodiment of this application. As shown in FIG. 9, the location information of the access request in the first cache queue may be recorded via a pointer (doorbell), to obtain a pointer (that is, a doorbell address of the first cache queue) of the first cache queue. The first instruction may include the pointer of the
20 first cache queue. For example, before step S4 is performed, the pointer of the first cache queue may be "3", indicating that a previously written access request is written into a 3rd entry of the first cache queue. After step S4 is performed, the access request is newly written into a 4th entry of the first cache queue. In this case, the pointer of the first cache queue may be updated to "4". In step S5, the processor may generate the first instruction, the pointer of the first cache queue included
25 in the first instruction may be "4". The processor may send, to the storage device to be accessed, the first instruction including the pointer "4" of the first cache queue.

[0093] Step S6: The storage device obtains the access request from the first cache queue and executes the access request according to the instruction of the first instruction.

[0094] FIG. 10 shows an example in which the storage device obtains the access request and
30 executes the access request according to an embodiment of this application.

[0095] For example, as shown in FIG. 10, the storage device receives the first instruction from the processor, and the first instruction instructs the storage device to obtain the access request from the first cache queue and execute the access request. That pointer information of the first cache queue included in the first instruction is "4" is used as an example. According to the instruction of
35 the first instruction, the storage device obtains the access request from a location of the 4th entry

in the first cache queue. With reference to related descriptions of step S5, the entry converted from the access request may be obtained.

[0096] The storage device may determine the type of the access request based on the operation code in the entry obtained through conversion. The storage device may determine, based on the scatter gather list in the entry obtained through conversion, the address (that is, the access memory address) of the object for data transmission in the PCIe point-to-point transmission mode. The storage device may execute the access request based on the type of the access request and the access memory address.

[0097] A manner in which the storage device executes the access request may be writing data into the memory of the data processing unit or reading data from the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode. When the type of the access request is the read request, the manner in which the storage device executes the access request may be reading data from the storage device based on the access request, and writing, into the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode, the data read from the storage device, for example, writing the data into the data page region of the memory of the data processing unit. When the type of the access request is the write request, the manner in which the storage device executes the access request may be obtaining, from the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode, data to be written into the storage device. For example, data may be obtained from the data page region of the memory of the data processing unit, and the obtained data is written into the storage device.

[0098] The access request may further include information for accessing a specific storage region of the storage device, for example, an identifier of a specific storage region of the storage device. When the type of the access request is the read request, data stored in a storage region corresponding to an identifier of a storage region included in the access request may be read from the storage device based on the access request. When the type of the access request is the write request, the obtained data may be written into the storage region that is in the storage device and that corresponds to the identifier of the storage region included in the access request.

[0099] Step S7: The storage device writes the execution result of the access request into the second cache queue of the processor.

[00100] The access request is generated by software such as the distributed storage software run by the data processing unit. Therefore, corresponding to the access request, the execution result of the access request may be transmitted to the software that generates the access request, to notify the software that generates the access request of an execution status of the access request generated by the software. There may be two types of the execution result of the access request. One is that

execution of the access request has been completed, and the other is that execution of the access request has been not completed.

[00101] FIG. 11 shows an example in which the storage device writes the execution result of the access request into the second cache queue according to an embodiment of this application. As shown in FIG. 11, the storage device may convert the execution result of the access request into a form of a completion queue (the second cache queue) entry (completion queue entry, CQE). The entry obtained through conversion may include an identifier and the like. Different types of the identifier may correspond to different execution results. For example, an identifier "1" may indicate that an entry having the identifier is an entry indicating that execution of the access request has been completed, and an identifier "0" may indicate that an entry having the identifier is an entry indicating that execution of the access request has not been completed. The second cache queue may include a plurality of entries. When the execution result of the access request is written into the second cache queue of the processor, the entry obtained through conversion based on the execution result of the access request may be written into a location of a specific entry (for example, an entry at a tail of the queue) in the second cache queue of the processor in the direct memory access (direct memory access, DMA) manner.

[00102] When each host-side storage device corresponds to one first cache queue, each host-side storage device may further correspond to one second cache queue. In this case, each second cache queue may also include an identifier of a storage device corresponding to the second cache queue. When the execution result of the access request is written into the second cache queue of the processor, the second cache queue with the same identifier may be found from a plurality of second cache queues based on the identifier included in the access request, and then the execution result of the access request is written into the second cache queue.

[00103] Step S8: The data processing unit obtains the execution result of the access request from the second cache queue of the processor.

[00104] FIG. 12 shows an example in which the data processing unit obtains the execution result of the access request according to an embodiment of this application.

[00105] For example, in step S7, the storage device writes the execution result of the access request into the second cache queue of the processor, and during writing, the data processing unit is not notified of the write operation. Therefore, for the data processing unit, an update time of the second cache queue of the processor is unknown. To obtain the execution result of the access request, as shown in FIG. 12, the data processing unit may be set to monitor, at intervals of a preset time period, whether the second cache queue of the processor is updated. When detecting that an execution result of a newly written access request appears in the second cache queue, the data processing unit obtains the execution result of the access request. The data processing unit may

transmit the execution result to the software that generates the access request. The data processing unit does not need to cache the execution result of the access request, so that occupation of a storage space of the data processing unit can be reduced.

[00106] FIG. 13 shows another example working procedure of a computing device according to an embodiment of this application. As shown in FIG. 13, for steps S1 to S6, refer to related descriptions of steps S1 to S6 in FIG. 4, and a part related to the second cache queue is deleted. Details are not described herein again.

[00107] After step S6 is completed, the computing device may perform step S9: A storage device writes an execution result of an access request into a data processing unit in a PCIe point-to-point transmission mode. FIG. 14 shows an example in which the storage device writes the execution result of the access request into the data processing unit according to an embodiment of this application.

[00108] For example, in step S6, after executing the access request, the storage device may generate one execution result of the access request. As shown in FIG. 14, when the data processing unit has a function of caching the execution result of the access request, the execution result of the access request may be directly written into the data processing unit in the PCIe point-to-point transmission mode. For example, a storage region may be spared in the storage space of the data processing unit, to cache the execution result of the access request. It may be set that interruption is generated to notify the data processing unit when stored content of the region is updated, or the data processing unit may be set to poll the region to monitor whether the stored content is updated. The interruption and polling may be implemented based on the conventional technology. When an execution result of a newly written access request enables the stored content to be updated, the data processing unit may find the execution result of the newly written access request, and transmit the execution result to software that generates the access request. The execution result of the access request does not need to be transmitted via a processor, so that efficiency of obtaining the execution result of the access request by the data processing unit can be improved.

[00109] FIG. 15 shows a schematic diagram of an example working procedure of a data access method according to an embodiment of this application.

[00110] As shown in FIG. 15, this embodiment of this application proposes the data access method. The method is performed by a data processing unit connected to a processor, and the method includes steps S10 and S11.

[00111] Step S10: Write an access request into a first cache queue of the processor, where the access request includes an identifier of a storage device to be accessed, and the storage device is connected to the processor and not directly connected to the data processing unit.

[00112] Step S11: Send, in a peripheral component interconnect express (PCIe) point-to-point

transmission mode, a first instruction to the storage device to which the identifier of the storage device points, where the first instruction includes location information of the access request in the first cache queue, and the first instruction instructs the storage device to obtain the access request from the first cache queue and execute the access request.

5 **[00113]** For an example implementation of step S10, refer to the foregoing and related descriptions of step S4 in FIG. 4 and FIG. 13. For an example implementation of step S11, refer to the foregoing and related descriptions of step S5 in FIG. 4 and FIG. 13.

10 **[00114]** According to the data access method in this embodiment of this application, the data processing unit is first used to write, into the first cache queue of the processor, the access request including the identifier of the storage device to be accessed, so that the access request may be stored in the first cache queue of the processor. Then the first instruction is sent, in the PCIe point-to-point transmission mode, to the storage device to which the identifier of the storage device points, so that the storage device to be accessed can receive the first instruction. The first instruction includes the location information of the access request in the first cache queue, and the first instruction instructs the storage device to obtain the access request from the first cache queue and execute the access request, so that after receiving the first instruction, the storage device can obtain the access request from a corresponding location in the first cache queue and execute the access request according to an instruction of the first instruction. In this manner, when the storage device is connected to the processor and not directly connected to the data processing unit, the data processing unit does not need to access the storage device via the processor of a host, and can directly access the storage device connected to the processor, so that resources of the host are not occupied. In addition, this access manner can simplify a hardware connection manner and reduce hardware design complexity.

25 **[00115]** In a possible implementation, the access request further includes a type of the access request and an access memory address, the access memory address is an address allocated by the processor to a memory of the data processing unit, the type of the access request indicates that the access request is a read request or a write request, and the access memory address is an address accessible to the storage device. For the access request, refer to the example of the access request in the foregoing and related descriptions in FIG. 8.

30 **[00116]** In this manner, when obtaining the access request from the first cache queue, the storage device may determine the type of the access request and address information of an access object, so that whether to perform a read operation or a write operation can be determined based on the type of the access request, and when executing the access request, data access by the data processing unit to the storage device of a corresponding operation type is completed based on the address information of the access object.

35

[00117] In a possible implementation, the method further includes: obtaining an execution result of the access request from a second cache queue of the processor. For an example implementation, refer to the foregoing and related descriptions of step S8 in FIG. 4 and the foregoing and related descriptions in FIG. 12.

5 **[00118]** Because the data processing unit is connected to the processor, the data processing unit can obtain the execution result of the access request from the second cache queue of the processor, to determine an execution status of the access request. The data processing unit does not need to cache the execution result of the access request, so that occupation of a storage space of the data processing unit can be reduced.

10 **[00119]** In a possible implementation, the method further includes: receiving an execution result that is of the access request and that is transmitted by the storage device in the PCIe point-to-point transmission mode. For an example implementation, refer to the foregoing and related descriptions of step S9 in FIG. 13, and the foregoing and related descriptions in FIG. 14.

[00120] In this manner, the execution result of the access request does not need to be transmitted
15 by the storage device to the data processing unit via the processor, but is directly transmitted by the storage device to the data processing unit. This can improve transmission efficiency of the execution result of the access request.

[00121] In a possible implementation, the method further includes: transmitting the address of the memory of the data processing unit to the processor; and receiving the access memory address
20 from the processor. For an example implementation of transmitting the address of the memory of the data processing unit to the processor, refer to the foregoing and related descriptions of step S1 in FIG. 4 and FIG. 13, and the foregoing and related descriptions in FIG. 5. For an example implementation of receiving the access memory address from the processor, refer to the foregoing and related descriptions of step S2 in FIG. 4 and FIG. 13, and the foregoing and related descriptions
25 in FIG. 6.

[00122] In this manner, the data processing unit may obtain the access memory address accessible to the storage device, so that the access memory address can be used as an address for the data processing unit to transmit data in the PCIe point-to-point transmission mode and it is possible that the data processing unit and the storage device transmit data when the data processing
30 unit and the storage device are not directly connected to each other.

[00123] In a possible implementation, the method further includes: receiving addresses of the first cache queue and the second cache queue that are sent by the processor, where the address of the first cache queue is used to write the access request into the first cache queue of the processor, and the address of the second cache queue is used to obtain the execution result of the access
35 request from the second cache queue. For an example implementation, refer to the foregoing and

related descriptions of step S3 in FIG. 4, and the foregoing and related descriptions in FIG. 7.

[00124] In this manner, the data processing unit can find the first cache queue based on the received address of the first cache queue, and when writing the access request into the first cache queue, the data processing unit may directly write the access request. The data processing unit can find the second cache queue based on the received address of the second cache queue, and when obtaining the execution result of the access request from the second cache queue, the data processing unit can directly obtain the execution result of the access request. Therefore, real-time performance and accuracy of writing the access request and obtaining the access request result are ensured.

[00125] FIG. 16 shows a schematic diagram of an example working procedure of a data access method according to an embodiment of this application.

[00126] As shown in FIG. 16, this application proposes the data access method. The method is applied to a computing device. The computing device includes a processor, a storage device connected to the processor, and a data processing unit connected to the processor and not directly connected to the storage device. The method includes steps S12 to S14.

[00127] Step S12: The data processing unit writes an access request into a first cache queue of the processor, where the access request includes an identifier of the storage device to be accessed.

[00128] Step S13: The data processing unit sends, in a peripheral component interconnect express (PCIe) point-to-point transmission mode, a first instruction to the storage device to which the identifier of the storage device points, where the first instruction instructs a location of the access request in the first cache queue.

[00129] Step S14: The storage device obtains the access request from the first cache queue and executes the access request according to the first instruction.

[00130] For an example implementation of step S12, refer to the foregoing and related descriptions of step S4 in FIG. 4 and FIG. 13. For an example implementation of step S13, refer to the foregoing and related descriptions of step S5 in FIG. 4 and FIG. 13. For an example implementation of step S14, refer to the foregoing and related descriptions of step S6 in FIG. 4 and FIG. 13.

[00131] According to the data access method in this embodiment of this application, the data processing unit is first used to write, into the first cache queue of the processor, the access request including the identifier of the storage device to be accessed, so that the access request may be stored in the first cache queue of the processor. Then the data processing unit is used to send, in the PCIe point-to-point transmission mode, the first instruction to the storage device to which the identifier of the storage device points, so that the storage device to be accessed can receive the first instruction. The first instruction includes location information of the access request in the first

cache queue, and the first instruction instructs the storage device to obtain the access request from the first cache queue and execute the access request, so that after receiving the first instruction, the storage device can obtain the access request from a corresponding location in the first cache queue and execute the access request according to an instruction of the first instruction. In this manner, in the computing device, when the storage device is connected to the processor and not directly connected to the data processing unit, the computing device does not need to access the storage device via the processor of a host, and can directly access the storage device connected to the processor, so that resources of the host are not occupied. In addition, this access manner can simplify a hardware connection manner and reduce hardware design complexity.

[00132] In a possible implementation, the access request further includes an access memory address, the access memory address is an address allocated by the processor to a memory of the data processing unit, and the access memory address is an address accessible to the storage device. In step S14, executing the access request includes: The storage device writes data into the memory of the data processing unit or reads data from the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode.

[00133] For the access memory address, refer to the example of the access memory address in the foregoing and related descriptions in FIG. 6. For an example implementation in which the storage device writes data into the memory of the data processing unit or reads data from the memory of the data processing unit, refer to the foregoing and related descriptions of step S6 in FIG. 4 and FIG. 13.

[00134] In this manner, the storage device can find an access object, that is, the memory of the data processing unit, based on the access memory address, so that the access request can be executed to write data into the memory of the data processing unit or read data from the memory of the data processing unit, to complete access by the memory of the data processing unit to the storage device.

[00135] In a possible implementation, the access request further includes a type of the access request, and the type of the access request indicates that the access request is a read request or a write request. When the access request is the read request, executing the access request includes: reading data from the storage device based on the access request, and writing, into the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode, the data read from the storage device. When the access request is the write request, executing the access request includes: obtaining, from the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode, data to be written into the storage device, and writing the obtained data into the storage device.

[00136] For the type of the access request, refer to the example of the type of the access request

in the foregoing and the related descriptions in FIG. 8. When the access request is the read request or the write request, for an example implementation of the execution of the access request, refer to the foregoing and related descriptions of step S6 in FIG. 4 and FIG. 13.

[00137] When the access request is the read request, the storage device can write data into the memory of the data processing unit, which is equivalent to reading the storage device by the memory of the data processing unit. When the access request is the write request, the storage device can read data from the memory of the data processing unit, which is equivalent to writing the storage device by the memory of the data processing unit. In this manner, corresponding to different types of the access request, the storage device can respond to different operations that meet a requirement of the access request.

[00138] In a possible implementation, the method further includes: The data processing unit sends the address of the memory of the data processing unit to the processor. The processor allocates the access memory address to the memory of the data processing unit based on the address of the memory of the data processing unit. The processor sends the access memory address to the data processing unit. For an example implementation, refer to the foregoing and related descriptions of steps S1 and S2 in FIG. 4 and FIG. 13.

[00139] In this manner, the processor may allocate the access memory address to the memory of the data processing unit, so that the access memory address can be used as an address for the data processing unit to transmit data in the PCIe point-to-point transmission mode and it is possible that the data processing unit and the storage device transmit data when the data processing unit and the storage device are not directly connected to each other. Allocating the access memory address to the memory of the data processing unit and sending the access memory address may be performed in a configuration phase, so that the access memory address is directly used after software in the data processing unit generates the access request. This can improve efficiency of executing the access request by the computing device.

[00140] In a possible implementation, the method further includes: The storage device writes an execution result of the access request into a second cache queue of the processor. The data processing unit obtains the execution result of the access request from the second cache queue of the processor. For an example implementation, refer to the foregoing and related descriptions of steps S7 and S8 in FIG. 4.

[00141] In a possible implementation, the method further includes: The data processing unit receives an execution result that is of the access request and that is transmitted by the storage device in the PCIe point-to-point transmission mode. For an example implementation, refer to the foregoing and related descriptions of step S9 in FIG. 13.

[00142] In a possible implementation, the method further includes: receiving addresses of the

first cache queue and the second cache queue that are sent by the processor, where the address of the first cache queue is used to write the access request into the first cache queue of the processor, and the address of the second cache queue is used to obtain the execution result of the access request from the second cache queue. For an example implementation, refer to the foregoing and related descriptions of step S3 in FIG. 4, and the foregoing and related descriptions in FIG. 7.

[00143] In this manner, the data processing unit can find the first cache queue based on the received address of the first cache queue, so that the data processing unit can accurately and quickly perform operations of writing the access request into the first cache queue and obtaining the execution result of the access request from the second cache queue. Creation of the first cache queue and the second cache queue and sending of the addresses of the first cache queue and the second cache queue may be performed in the configuration phase, so that the addresses of the first cache queue and the second cache queue are directly used after the software in the data processing unit generates the access request. This can improve the efficiency of executing the access request by the computing device.

[00144] FIG. 17 shows a schematic structural diagram of a computing device according to an embodiment of this application.

[00145] As shown in FIG. 17, this embodiment of this application further provides the computing device, including:

a processor 101;

a storage device 102, connected to the processor 101; and

a data processing unit 103, connected to the processor 101 and not directly connected to the storage device 102, and configured to: write an access request into a first cache queue of the processor 101, where the access request includes an identifier of the storage device 102 to be accessed; and send, in a peripheral component interconnect express (PCIe) point-to-point transmission mode, a first instruction to the storage device 102 to which the identifier of the storage device points, where the first instruction instructs a location of the access request in the first cache queue. For an example implementation, refer to the foregoing and related descriptions of steps S4 and S5 in FIG. 4 and FIG. 13, and related descriptions in FIG. 8 and FIG. 9.

[00146] The storage device 102 is configured to obtain the access request from the first cache queue and execute the access request according to the first instruction. For an example implementation, refer to the foregoing, related descriptions of step S6 in FIG. 4 and FIG. 13, and related descriptions in FIG. 10.

[00147] For the processor 101, refer to the example of the central processing unit in FIG. 3. For the storage device 102, refer to the example of the storage device HDD/SSD and the NVMe SSD in FIG. 3. For the data processing unit 103, refer to the example of the data processing unit in FIG.

3.

[00148] In a possible implementation, the access request further includes an access memory address, the access memory address is an address allocated by the processor 101 to a memory of the data processing unit 103, and the access memory address is an address accessible to the storage device 102.

[00149] The storage device 102 is configured to write data into the memory of the data processing unit 103 or read data from the memory of the data processing unit 103 based on the access memory address in the PCIe point-to-point transmission mode.

[00150] In a possible implementation, the access request further includes a type of the access request, and the type of the access request indicates that the access request is a read request or a write request.

[00151] When the access request is the read request, the storage device 102 is configured to: read data from the storage device 102 based on the access request, and write, into the memory of the data processing unit 103 based on the access memory address in the PCIe point-to-point transmission mode, the data read from the storage device 102.

[00152] When the access request is the write request, the storage device 102 is configured to: obtain, from the memory of the data processing unit 103 based on the access memory address in the PCIe point-to-point transmission mode, data to be written into the storage device 102, and write the obtained data into the storage device 102.

[00153] In a possible implementation, the processor 101 is configured to:
receive the address that is of the memory of the data processing unit 103 and that is sent by the data processing unit 103;
allocate the access memory address to the memory of the data processing unit 103 based on the address of the memory of the data processing unit 103; and
send the access memory address to the data processing unit 103.

[00154] In a possible implementation, the data processing unit 103 is configured to obtain an execution result of the access request from a second cache queue of the processor 101.

[00155] In a possible implementation, the data processing unit 103 is configured to receive an execution result that is of the access request and that is transmitted by the storage device 102 in the PCIe point-to-point transmission mode.

[00156] The computing device may include at least one of a desktop computer, a laptop computer, a handheld computer, a notebook computer, an ultra-mobile personal computer (ultra-mobile personal computer, UMPC), a netbook, a personal digital assistant (personal digital assistant, PDA), an augmented reality (augmented reality, AR) device, a virtual reality (virtual reality, VR) device, an artificial intelligence (artificial intelligence, AI) device, a wearable device,

a vehicle-mounted device, a smart home device, a smart city device, and a server device. A specific type of the computing device is not specially limited in this embodiment of this application.

[00157] The computing device may include the processor 101, the storage device 102, and the data processing unit 103. It may be understood that, a structure illustrated in this embodiment of this application does not constitute a specific limitation on the computing device. In some other embodiments in this application, the computing device may include more or fewer components than those shown in the figure, or some components may be combined, or some components may be separated, or different component arrangement may be used. The components shown in the figure may be implemented by hardware, software, or a combination of software and hardware.

[00158] An embodiment of this application provides a non-volatile computer-readable storage medium, where the non-volatile computer-readable storage medium stores computer program instructions. When the computer program instructions are executed by a processor, the foregoing method is implemented.

[00159] An embodiment of this application provides a computer program product including computer-readable code or a non-volatile computer-readable storage medium carrying the computer-readable code. When the computer-readable code is run in a processor of an electronic device, the processor in the electronic device performs the foregoing method.

[00160] The computer-readable storage medium may be a tangible device that may retain and store instructions for use by an instruction execution device. The computer-readable storage medium may be, for example, but is not limited to, an electrical storage device, a magnetic storage device, an optical storage device, an electromagnetic storage device, a semiconductor storage device, or any suitable combination thereof. More specific examples (a non-exhaustive list) of the computer-readable storage medium include: a portable computer disk, a hard disk, a random access memory (Random Access Memory, RAM), a read-only memory (Read Only Memory, ROM), an erasable programmable read-only memory (erasable Programmable Read-Only-Memory, EPROM, or flash memory), a static random access memory (Static Random Access Memory, SRAM), a portable compact disc read-only memory (Compact Disc Read-Only Memory, CD-ROM), a digital versatile disc (Digital Video Disc, DVD), a memory stick, a floppy disk, a mechanical coding device, for example, a punching card or a groove protrusion structure that stores instructions, and any suitable combination thereof.

[00161] The computer-readable program instructions or code described herein may be downloaded from the computer-readable storage medium to each computing/processing device, or downloaded to an external computer or an external storage device over a network, such as the Internet, a local area network, a wide area network, and/or a wireless network. The network may include a copper transmission cable, optical fiber transmission, wireless transmission, a router, a

firewall, a switch, a gateway computer, and/or an edge server. A network adapter card or a network interface in each computing/processing device receives computer-readable program instructions from the network, and forwards the computer-readable program instructions for storage in the computer-readable storage medium in each computing/processing device.

5 **[00162]** The computer program instructions used to perform operations in this application may be assembly instructions, instruction set architecture (Instruction Set Architecture, ISA) instructions, machine instructions, machine-related instructions, microcode, firmware instructions, status setting data, or source code or target code written in one or more programming languages or any combination thereof. The programming languages include an object-oriented programming
10 language such as Smalltalk and C++, and a conventional procedural programming language such as a "C" language or a similar programming language. The computer-readable program instructions may be executed entirely on a user computer, partly on the user computer, as a stand-alone software package, partly on the user computer and partly on a remote computer, or entirely on the remote computer or a server. When the remote computer is involved, the remote computer
15 may be connected to the user computer over any type of network, including a local area network (Local Area Network, LAN) or a wide area network (Wide Area Network, WAN), or may be connected to an external computer (for example, connected via an Internet service provider over the Internet). In some embodiments, an electronic circuit, for example, a programmable logic circuit, a field programmable gate array (Field Programmable Gate Array, FPGA), or a
20 programmable logic array (Programmable Logic Array, PLA), is customized via status information of the computer-readable program instructions. The electronic circuit may execute the computer-readable program instructions, to implement various aspects of this application.

[00163] The various aspects of this application are described herein with reference to the flowcharts and/or the block diagrams of the method, the apparatus (system), and the computer
25 program product according to embodiments of this application. It should be understood that each block in the flowcharts and/or the block diagrams and a combination of blocks in the flowcharts and/or the block diagrams may all be implemented by the computer-readable program instructions.

[00164] The flowcharts and the block diagrams in the accompanying drawings illustrate system architectures, functions, and operations of possible implementations of apparatuses, systems,
30 methods, and computer program products according to a plurality of embodiments of this application. In this regard, each block in the flowcharts or the block diagrams may represent a module, a program segment, or a part of the instructions, where the module, the program segment, or the part of the instructions includes one or more executable instructions for implementing a specified logical function. In some alternative implementations, the functions marked in the blocks
35 may also occur in a sequence different from that marked in the accompanying drawings. For

example, two consecutive blocks may actually be executed substantially in parallel, and sometimes may be executed in a reverse order, depending on a function involved.

[00165] It should also be noted that each block in the block diagrams and/or the flowcharts and the combination of blocks in the block diagrams and/or the flowcharts may be implemented by hardware (for example, a circuit or an ASIC (application-specific integrated circuit)) that performs a corresponding function or action, or may be implemented by a combination of hardware and software, for example, firmware.

[00166] Although the present invention is described with reference to embodiments, in a process of implementing the present invention that claims protection, a person skilled in the art may understand and implement another variation of the disclosed embodiments by viewing the accompanying drawings, the disclosed content, and the accompanying claims. In the claims, "comprising" (comprising) does not exclude another component or another step, and "a" or "one" does not exclude a case of multiple. A single processor or another unit may implement several functions enumerated in the claims. Some measures are recorded in dependent claims that are different from each other, but this does not mean that these measures cannot be combined to produce a better effect.

[00167] Embodiments of this application are described above. The foregoing descriptions are examples, are not exhaustive, and are not limited to the disclosed embodiments. Many modifications and changes are clear to a person of ordinary skill in the art without departing from the scope of the described embodiments. The selection of terms used in this specification is intended to best explain principles of embodiments, practical applications, or improvements to technologies in the market, or to enable another person of ordinary skill in the art to understand embodiments disclosed herein.

CLAIMS

What is claimed is:

1. A data access method, wherein the method is performed by a data processing unit connected to a processor, and the method comprises:

5 writing an access request into a first cache queue of the processor, wherein the access request comprises an identifier of a storage device to be accessed, and the storage device is connected to the processor and not directly connected to the data processing unit; and

 sending, in a peripheral component interconnect express (PCIe) point-to-point transmission mode, a first instruction to the storage device to which the identifier of the storage device points,
10 wherein the first instruction comprises location information of the access request in the first cache queue, and the first instruction instructs the storage device to obtain the access request from the first cache queue and execute the access request.

2. The method according to claim 1, wherein the access request further comprises a type of the access request and an access memory address, the access memory address is an address
15 allocated by the processor to a memory of the data processing unit, the type of the access request indicates that the access request is a read request or a write request, and the access memory address is an address accessible to the storage device.

3. The method according to claim 1 or 2, wherein the method further comprises:

 obtaining an execution result of the access request from a second cache queue of the processor.

20 4. The method according to claim 1 or 2, wherein the method further comprises:

 receiving an execution result that is of the access request and that is transmitted by the storage device in the PCIe point-to-point transmission mode.

5. The method according to any one of claims 2 to 4, wherein the method further comprises:

 transmitting the address of the memory of the data processing unit to the processor; and

25 receiving the access memory address from the processor.

6. A data access method, wherein the method is applied to a computing device, the computing device comprises a processor, a storage device connected to the processor, and a data processing unit connected to the processor and not directly connected to the storage device, and the method comprises:

30 writing, by the data processing unit, an access request into a first cache queue of the processor, wherein the access request comprises an identifier of the storage device to be accessed;

 sending, by the data processing unit in a peripheral component interconnect express (PCIe) point-to-point transmission mode, a first instruction to the storage device to which the identifier of the storage device points, wherein the first instruction instructs a location of the access request in

the first cache queue; and

obtaining, by the storage device, the access request from the first cache queue and executing the access request according to the first instruction.

5 7. The method according to claim 6, wherein the access request further comprises an access memory address, the access memory address is an address allocated by the processor to a memory of the data processing unit, and the access memory address is an address accessible to the storage device; and

10 the executing the access request comprises: writing, by the storage device, data into the memory of the data processing unit or reading data from the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode.

8. The method according to claim 7, wherein the access request further comprises a type of the access request, and the type of the access request indicates that the access request is a read request or a write request; and

15 when the access request is the read request, the executing the access request comprises: reading data from the storage device based on the access request, and writing, into the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode, the data read from the storage device; or

20 when the access request is the write request, the executing the access request comprises: obtaining, from the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode, data to be written into the storage device, and writing the obtained data into the storage device.

9. The method according to claim 7 or 8, wherein the method further comprises:

sending, by the data processing unit, the address of the memory of the data processing unit to the processor;

25 allocating, by the processor, the access memory address to the memory of the data processing unit based on the address of the memory of the data processing unit; and

sending, by the processor, the access memory address to the data processing unit.

10. The method according to any one of claims 6 to 9, wherein the method further comprises:

30 writing, by the storage device, an execution result of the access request into a second cache queue of the processor; and

obtaining, by the data processing unit, the execution result of the access request from the second cache queue of the processor.

11. The method according to any one of claims 6 to 9, wherein the method further comprises:

35 receiving, by the data processing unit, an execution result that is of the access request and that is transmitted by the storage device in the PCIe point-to-point transmission mode.

12. A computing device, comprising:

a processor;

a storage device, connected to the processor; and

a data processing unit, connected to the processor and not directly connected to the storage device, and configured to:

write an access request into a first cache queue of the processor, wherein the access request comprises an identifier of the storage device to be accessed; and

send, in a peripheral component interconnect express (PCIe) point-to-point transmission mode, a first instruction to the storage device to which the identifier of the storage device points, wherein the first instruction instructs a location of the access request in the first cache queue, wherein

the storage device is configured to obtain the access request from the first cache queue and execute the access request according to the first instruction.

13. The device according to claim 12, wherein the access request further comprises an access memory address, the access memory address is an address allocated by the processor to a memory of the data processing unit, and the access memory address is an address accessible to the storage device; and

the storage device is configured to write data into the memory of the data processing unit or read data from the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode.

14. The device according to claim 13, wherein the access request further comprises a type of the access request, and the type of the access request indicates that the access request is a read request or a write request; and

when the access request is the read request, the storage device is configured to: read data from the storage device based on the access request, and write, into the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode, the data read from the storage device; or

when the access request is the write request, the storage device is configured to: obtain, from the memory of the data processing unit based on the access memory address in the PCIe point-to-point transmission mode, data to be written into the storage device, and write the obtained data into the storage device.

15. The device according to claim 13 or 14, wherein the processor is configured to:

receive the address that is of the memory of the data processing unit and that is sent by the data processing unit;

allocate the access memory address to the memory of the data processing unit based on the

address of the memory of the data processing unit; and

send the access memory address to the data processing unit.

16. The device according to any one of claims 12 to 15, wherein the data processing unit is configured to:

5 obtain an execution result of the access request from a second cache queue of the processor.

17. The device according to any one of claims 12 to 15, wherein the data processing unit is configured to:

receive an execution result that is of the access request and that is transmitted by the storage device in the PCIe point-to-point transmission mode.

10