WRITE REQUEST PROCESSING METHOD, PROCESSOR, AND COMPUTER

TECHNICAL FIELD

Embodiments of the present invention relate to the field of computers, and more specifically, to a write request processing method, a processor, and a computer.

BACKGROUND

A cluster computer system generally consists of multiple computers, respective operating systems run on the multiple computers respectively, and the multiple computers can work independently. The multiple computers are interconnected by using an input/output (I/O) bus such as an Ethernet bus, a Peripheral Component Interconnect Express (PCIe) bus, or an Infiniband bus, which can implement communication and data exchange between the computers.

The cluster computer system is widely used in the field of high-end storage. For example, for each high-end storage device, a dual-controller or multi-controller storage area network (Storage Area Network, SAN) system is generally used, and each controller in the dual-controller or multi-controller SAN system may be viewed as an independent computer. A respective operating system runs on each controller, and a respective memory is disposed on each controller. The controllers are interconnected by using an IO bus, forming the cluster computer system. The dual-controller or multi-controller SAN system used in the high-end storage device can ensure data storage reliability. With reference to FIG. 1, the following provides detailed descriptions by using a dual-controller SAN system based on PCIe interconnection as an example.

As shown in FIG. 1, the dual-controller SAN system includes a controller A and a controller B, where both the controller A and the controller B include a CPU, a DDR, and a front-end bus, and respective operating systems run on the controller A and the controller B respectively. The controller A is connected to the controller B by using a PCIe bus, and domain address conversion and data exchange are implemented between two computers by using a PCIe non-transparent bridge (Non-Transparent Bridge, NTB).

When the controller A receives, by using the front-end bus, an external write request (which may also be referred to as write IO or a write IO request) used to store data, a CPU 0 of the controller A

first writes the data into a DDR 0. Then, the CPU 0 reads the data from the DDR 0, encapsulates the data into data that satisfies a PCIe format, and sends data obtained by means of encapsulation to the controller B by using a mirror channel between the two controllers that is formed by using a PCIe bus. A CPU 1 of the controller B obtains the data by means of decapsulation and then writes the data into a DDR 1.

The foregoing is an entire mirroring operation process in the dual-controller SAN system. Such a mirroring operation is equivalent to backing, in the controller B, up data in the controller A. When a fatal error occurs on the controller A and a breakdown is caused, the controller B can still work normally, and that data in a memory of the controller A is lost due to the breakdown of the controller A is avoided. In this case, the entire system can still run and work normally, improving reliability of a storage device. However, it can be seen, from the mirroring process described above, that during the entire mirroring operation, a direct memory access (DMA) operation needs to be performed twice (for the first time, reading data from the DDR 0, and for the second time, writing data into the DDR 1). In addition, an IO access between devices interconnected under a PCIe (Ethernet, Infiniband, or the like) network protocol is bound to bring a large quantity of software protocol overheads (for example, data encapsulation and decapsulation), which causes high mirroring operation latency and relatively poor read/write operations per second (Input/Output Operations Per Second, IOPS) performance of the system.

SUMMARY

Embodiments of the present invention provide a write request processing method and a processor, and a computer, which can reduce data mirroring operation latency in a cluster computer system and improve IOPS of the system.

According to a first aspect, a write request processing method is provided, where the method is applied to a first computer, the first computer is connected to a second computer, respective operating systems run on the first computer and the second computer respectively, the first computer includes a first processor, the second computer includes a second processor, both the first processor and the second processor include a system bus interface, and the system bus interface of the first processor is connected to the system bus interface of the second processor by using a system bus; and the method includes: receiving, by the first processor, a write request, where the write request includes to-be-written target data and a write address of the target data; determining, by the first processor, that the write address is located in a first memory address space; and writing, by the first processor, the target data in the write request into the first memory address space, and sending the target data in the write request to a second processor by using the system bus.

processor and sending the target data to the second processor by the first processor by using the system bus, which may be performed at the same time, or may be performed successively.

In the prior art, a system bus is generally used to implement a connection of functional components inside a computer, and the system bus is one of internal buses of the computer and has features of high bandwidth and low latency. In this embodiment of the present invention, processors that respectively belong to two different computers are connected in a cluster computer system together by using a system bus. The system bus is used to implement a mirroring operation of to-be-written data, without requiring multiple DMA operations like that in the prior art or without complex protocol–based conversion, which reduces mirroring operation latency and improves IOPS performance of a system. Specifically, in the prior art, an IO device is used to transport a mirror packet between two computers. Therefore, mirror data (corresponding to the foregoing target data) needs to be taken out of a memory first; then, the target data is sent to the IO device (using a network adapter as an example); next, the network adapter encapsulates the mirror data into a packet by using a software driver of the network adapter and sends the packet to a receive end; the receive end decapsulates the packet after receiving the packet, and then writes the packet into a memory of the receive end. In the entire process, multiple memory read and write operations are required, and complex protocol–based encapsulation and decapsulation are involved. In this embodiment of the present invention, the target data is directly sent by the first processor to the second processor by using the system bus, which eliminates a process of reading the data from the memory. In addition, although the target data may also be encapsulated in a packet (for example, a mirror packet described in the following) and be sent by using the system bus, the system bus is one of internal buses, and an encapsulation form of a packet used on the system bus may also be designed to be relatively simple. For example, a data write packet and a data mirror packet may be distinguished by changing some fields in the write packet, without a need of involving complex protocols and encapsulation forms like an IO device (for example, a network adapter).

Optionally, in an embodiment, the processor may include one or more system bus interfaces.

Optionally, in an embodiment, the system bus may be a bus supporting cache coherence (CC). For example, the system bus may be referred to as a CC bus. In an embodiment, the system bus may be at least one of a HyperTransport (HT) bus or a quick path interconnect (Quick Path Interconnect, QPI) bus.

Optionally, in an embodiment, the method shown in FIG. 3 may be applied to the field of storage devices. In the field of storage devices, a cluster computer system may be a dual-controller or multi-controller system. Either computer of the first computer and the second computer is equivalent to a controller in the field of storage devices.

Step 4: The system bus interface of the second processor sends the mirror packet to a non-transparent bridge of the second processor.

Step 5: The non-transparent bridge of the second processor converts the mirror packet to a write packet and sends the write packet to a memory controller by using an internal bus of the second processor, to complete a corresponding memory write operation.

Step 6: The memory controller of the second processor returns a mirror write completion packet to the non-transparent bridge of the second processor after completing the memory write operation.

Step 7: The non-transparent bridge of the second processor modifies the address field 0X0A_1234 in the mirror write completion packet to 0X04_1234 according to the mirror relationship between the memory address spaces, and sends an updated packet to the system bus interface of the second processor.

Step 8: The system bus interface of the second processor sends the mirror completion packet to the system bus interface of the first processor.

Step 9: The system bus interface of the first processor sends the mirror completion packet to the non-transparent bridge of the first processor.

Step 10: The non-transparent bridge of the first processor sends, by using the internal bus of the first processor, the mirror completion packet to an initiator of the mirror packet, that is, the CA in step 1.

Step 11: The CA receives the mirror completion packet, determines that a mirroring operation is successfully completed, terminates the transaction, and releases a corresponding resource.

In addition, a timer may be disposed on the non-transparent bridge of the first processor. When the mirror completion packet has not been received yet when the timer expires, a mirror timeout packet may be sent to the CA to indicate a failure of the mirroring operation.

It can be seen from the foregoing procedure that, a function of the NTB may include memory address conversion, packet or packet format conversion, determining on abnormal termination of a mirror transaction, report interruption, and the like.

On a basis of introducing the mirror packet and the non-transparent bridge, optionally, as an embodiment, step 330 may include: generating, by the first processor, a mirror packet according to the write request, where the mirror packet includes the target data and the write address, and the mirror packet is used to write the target data into a mirror address, of the write address, in the second memory address space; sending, by the first processor, the mirror packet to a non-transparent bridge of the first processor; converting, by the non-transparent bridge of the first processor, the write address in the mirror packet to the mirror address according to a mirror relationship between the first memory address space and the second memory address space, to obtain an updated mirror packet; and sending, by the non-transparent bridge of the first processor,

In this embodiment of the present invention, complete logic about performing a mirroring operation on a system bus is added to the first processor. The logic ensures that the mirroring operation herein is to be terminated regardless of a success or failure of the mirroring operation, which can avoid a system breakdown caused by a failure of a mirroring operation on a system bus. On this basis, brute-force surprise expansion of a system bus interface, for example, surprise expansion or surprise removal of the system bus interface, of a processor inside a computer can be further supported. Specifically, in the prior art, multiple processors in a stand-alone system are also connected by using a system bus. For example, in a non-uniform memory access architecture (NUMA) system supporting cache coherence in the prior art, each processor may access a local memory, or may access a non-local memory by using a system bus, that is, memories of the multiple processors may be shared. However, a single operating system runs on the memories of such a multi-processor system, and in essence, the multi-processor system is still a stand-alone system instead of a cluster computer system. The operation performed between the multiple processors by using the system bus is still regarded in essence as an operation inside the stand-alone system. All operations on the system bus of the stand-along system are completed by hardware. Once an error occurs during an operation on a system bus of a processor (for example, brute-force surprise expansion occurs on the system bus interface), the error is a hardware error. The hardware error causes a breakdown of the stand-alone system. To avoid a system breakdown caused by an operation error on the system bus, complete logic about performing a mirroring operation on the system bus is added to the first processor. The logic ensures that the mirroring operation herein is to be terminated regardless of a success or failure of the mirroring operation; in this way, a system breakdown due to incompletion of or an error during an operation on a system bus can be avoided.

With reference to FIG. 5 and FIG. 6, the following describes in detail a process of a data mirroring operation between computers, connected by using a system bus, in a cluster computer system by using an example in which a processor is a processor supporting a QPI technology and the system bus is a QPI bus (for a specific structure, refer to FIG. 4). It should be understood that this embodiment of the present invention imposes no limitation on a time sequence of steps in FIG. 5 and FIG. 6, all steps in FIG. 5 and FIG. 6 may be performed in a sequence different from that shown in FIG. 5 and FIG. 6, and possibly, not all steps in FIG. 5 and FIG. 6 are necessary to be performed. A triggering condition of the mirroring operation process described in FIG. 5 and FIG. 6 may be: a CA in a first computer discovers that a write address of a write request received by the CA is located in a mirror address space (corresponding to the foregoing first memory address space) of a memory of the first computer, where the mirror address space may be preconfigured.

Optionally, as an embodiment, the non-transparent bridge 840 is further configured to send a mirror timeout packet to the controller 820 when the mirror completion packet is not received from the another processor within the preset time; and the controller 820 is further configured to confirm a failure of the mirroring operation according to the mirror timeout packet, and terminate the mirror transaction.

Optionally, as an embodiment, the controller 820 is specifically configured to generate a mirror packet according to the write request, where the mirror packet includes the target data and the write address; and send the mirror packet to a non-transparent bridge of the processor, where the non-transparent bridge of the processor sends the mirror packet to a non-transparent bridge of the another processor by using the system bus, so that the non-transparent bridge of the another processor converts the write address in the mirror packet to a mirror address, of the write address, in the second memory address space, to obtain an updated mirror packet, and the another processor writes the target data into the mirror address according to the updated mirror packet.

Optionally, as an embodiment, the controller is specifically configured to generate a write packet for the target data according to the write request; and write the target data into the write address in the first memory address space according to the write packet.

Optionally, as an embodiment, the system bus is a QPI bus or an HT bus.

FIG. 9 is a schematic structural diagram of a processor according to an embodiment of the present invention. The processor 900 in FIG. 9 can implement all the steps, performed by the second processor, in FIG. 1 to FIG. 7. To avoid repetition, details are not described herein again. The processor 900 is located in a second computer connected to a first computer, respective operating systems run on the first computer and the second computer respectively, the first computer includes another processor, the processor 900 includes a system bus interface 910, a system bus interface of the another processor is connected to the system bus interface of the processor by using a system bus, a second memory address space of the second computer is a mirror address space of a first memory address space of the first computer, the processor 900 includes a controller 920, an internal bus 930, and a non-transparent bridge 940, the non-transparent bridge 940 is connected to the system bus interface 910 of the processor 900, and the controller 920 is connected to the non-transparent bridge 940 by using the internal bus 930.

The non-transparent bridge 940 is configured to receive target data from the another processor by using the system bus, where the target data is data to be written into a memory of the another processor, and a write address of the target data is located in the first memory address space; and write the target data into the second memory address space.

Optionally, as an embodiment, the mirroring module 1030 is further configured to generate a mirror packet according to the write request, where the mirror packet includes the target data and the write address; and send the mirror packet to a non-transparent bridge of the first processor, where the non-transparent bridge of the first processor sends the mirror packet to a non-transparent bridge of the second processor by using the system bus, so that the non-transparent bridge of the second processor converts the write address in the mirror packet to a mirror address, of the write address, in the second memory address space, to obtain an updated mirror packet, and the second processor writes the target data into the mirror address according to the updated mirror packet.

Optionally, as an embodiment, the mirroring module 1030 is specifically configured to generate a write packet for the target data according to the write request; and write the target data into the write address in the first memory address space according to the write packet.

Optionally, as an embodiment, the system bus is a QPI bus or an HT bus.

FIG. 11 is a schematic structural diagram of a computer according to an embodiment of the present invention. The computer 1100 in FIG. 11 is corresponding to the second computer in FIG. 1 to FIG. 7 and can implement all the steps performed by the second processor of the second computer. To avoid repetition, details are not described herein again. The computer 1100 is connected to another computer, respective operating systems run on the another computer and the computer 1100 respectively, the another computer includes a first processor, the computer 1100 includes a second processor, both the first processor and the second processor include a system bus interface, the system bus interface of the first processor is connected to the system bus interface of the second processor by using a system bus, and a second memory address space of the computer 1100 is a mirror address space of a first memory address space of the another computer.

The computer 1100 includes:

a mirroring module 1110, configured to receive target data from the first processor by using the system bus, where the target data is data to be written into a memory of the first processor, and a write address of the target data is located in the first memory address space; and

a write operation module 1120, configured to write the target data into the second memory address space.

In this embodiment of the present invention, processors of two different computers are connected together by using a system bus. The system bus is used to implement a mirroring operation of to-be-written data, without requiring multiple DMA operations like that in the prior art or without complex protocol–based conversion, which reduces mirroring operation latency and improves IOPS performance of a system.

processor by using the system bus, wherein the mirror packet comprises the target data and the write address; and

the writing, by the second processor, the target data into the second memory address space comprises:

converting, by the non-transparent bridge of the second processor, the mirror packet to a write packet for the target data, and converting the write address to a mirror address, of the write address, in the second memory address space; and

writing, by the non-transparent bridge of the second processor, the write packet into the mirror address in the second memory address space by using a memory controller of the second computer.

12. The method according to any one of claims 8 to 11, wherein the system bus is a quick path interconnect, QPI bus or HyperTransport, HT bus.

13. A computer, wherein the computer is connected to another computer, respective operating systems run on the computer and the another computer respectively, the computer comprises a first processor, the another computer comprises a second processor, both the first processor and the second processor comprise a system bus interface, the system bus interface of the first processor is connected to the system bus interface of the second processor by using a system bus, and a second memory address space of the another computer is a mirror address space of a first memory address space of the computer; and

the computer comprises:

a receiving module, configured to receive a write request, wherein the write request comprises to-be-written target data and a write address of the target data;

a determining module, configured to determine that the write address is located in the first memory address space; and

a mirroring module, configured to write the target data in the write request into the first memory address space, and send the target data in the write request to the second processor by using the system bus, so that the second processor writes the target data into the second memory address space.

14. The computer according to claim 13, wherein the mirroring module is specifically configured to generate a mirror packet according to the write request, wherein the mirror packet comprises the target data and the write address; send the mirror packet to a non-transparent bridge of the first processor; convert the write address in the mirror packet to a mirror address, of the write address, in the second memory address space according to a mirror relationship between the first memory address space and the second memory address space, to obtain an updated mirror packet; and send the updated mirror packet to a non-transparent bridge of the second processor by using the

system bus, so that the second processor writes the target data into the mirror address according to the updated mirror packet received by the non-transparent bridge of the second processor.

15. The computer according to claim 14, wherein the mirroring module is further configured to: send a mirror completion packet to the first processor after the non-transparent bridge of the first processor receives the mirror completion packet from the second processor within a preset time, wherein the mirror completion packet is corresponding to the mirror packet and is used to indicate that the target data has been written into the mirror address; and confirm a success of a mirroring operation for the target data according to the mirror completion packet, and terminate a mirror transaction corresponding to the mirroring operation.

16. The computer according to claim 15, wherein the mirroring module is further configured to: send a mirror timeout packet to the first processor when the non-transparent bridge of the first processor does not receive the mirror completion packet from the second processor within the preset time; and confirm a failure of the mirroring operation according to the mirror timeout packet, and terminate the mirror transaction.

17. The computer according to claim 13, wherein the mirroring module (1030) is further configured to generate a mirror packet according to the write request, wherein the mirror packet comprises the target data and the write address; and send the mirror packet to a non-transparent bridge of the first processor, wherein the non-transparent bridge of the first processor sends the mirror packet to a non-transparent bridge of the second processor by using the system bus, so that the non-transparent bridge of the second processor converts the write address in the mirror packet to a mirror address, of the write address, in the second memory address space, to obtain an updated mirror packet, and the second processor writes the target data into the mirror address according to the updated mirror packet.

18. The computer according to any one of claims 13 to 17, wherein the mirroring module (1030) is specifically configured to generate a write packet for the target data according to the write request; and write the target data into the write address in the first memory address space according to the write packet.

19. The computer according to any one of claims 13 to 18, wherein the system bus is a quick path interconnect, QPI bus or HyperTransport, HT bus.

20. A computer, wherein the computer is connected to another computer, respective operating systems run on the another computer and the computer respectively, the another computer comprises a first processor, the computer comprises a second processor, both the first processor and the second processor comprise a system bus interface, the system bus interface of the first processor is connected to the system bus interface of the second processor by using a system bus, and a

second memory address space of the computer is a mirror address space of a first memory address space of the another computer; and

the computer comprises:

a mirroring module(1110), configured to receive target data from the first processor by using the system bus, wherein the target data is data to be written into a memory of the first processor, and a write address of the target data is located in the first memory address space; and

a write operation module(1120), configured to write the target data into the second memory address space.

21. The computer according to claim 20, wherein the mirroring module is specifically configured to receive a mirror packet from the first processor by using the system bus, wherein the mirror packet comprises the target data and a mirror address, of the write address, in the second memory address space; and the write operation module is specifically configured to convert the mirror packet to a write packet for the target data, and write the write packet into the mirror address in the second memory address space by using a memory controller of the computer.

22. The computer according to claim 21, wherein the mirroring module (1110) is further configured to receive, from the memory controller, a write completion packet corresponding to the write packet, wherein the write completion packet comprises the mirror address; convert the mirror address to the write address according to a mapping relationship between the first memory address space and the second memory address space; and send, to the first processor, a mirror completion packet corresponding to the mirror packet, wherein the mirror completion packet comprises the write address, and the mirror completion packet is used to indicate that the target data has been written into the mirror address.

23. The computer according to claim 20, wherein the mirroring module (1110) is specifically configured to receive a mirror packet from the first processor by using the system bus, wherein the mirror packet comprises the target data and the write address; and the write operation module (1120) is specifically configured to convert the mirror packet to a write packet for the target data, and convert the write address to a mirror address, of the write address, in the second memory address space; and write the write packet into the mirror address in the second memory address space by using a memory controller of the computer.

24. The computer according to any one of claims 20 to 23, wherein the system bus is a quick path interconnect, QPI bus or HyperTransport, HT bus.