REMOTE MEMORY CALL METHOD, NETWORK INTERFACE CARD, ELECTRONIC DEVICE, AND STORAGE MEDIUM

[0001] This application claims priority to Chinese Patent Application No. 202311101531.5, filed with the China National Intellectual Property Administration on August 29, 2023 and entitled "REMOTE MEMORY CALL METHOD, NETWORK INTERFACE CARD, ELECTRONIC DEVICE, AND STORAGE MEDIUM", which is incorporated herein by reference in its entirety.

TECHNICAL FIELD

[0002] This application relates to the field of cloud computing technologies, and in particular, to a remote memory call method, a network interface card, an electronic device, and a storage medium.

BACKGROUND

[0003] Memory is an important component of a computing device, and is configured to temporarily store operational data of a processor and data exchanged with a hard disk or other external memories. In the computing device, all programs are run in the memory. The performance of the memory affects the performance of the computing device. When memory usage is excessive, the computing device is prone to stuttering and crashing, affecting the running of related services.

[0004] When a single computing device has limited memory, the utilization of the memory can be improved by reusing the memory of the device. However, constrained by existing memory capacity of the single computing device, the usage upper limit of the computing device remains unbreakable. Therefore, a new memory call mechanism needs to be provided.

SUMMARY

[0005] Embodiments of this application provide a remote memory call method, a network interface card, an electronic device, and a storage medium, to solve one or more of the foregoing technical problems.

[0006] According to a first aspect, this application provides a remote memory call method,

applied to an intermediate device connected to a plurality of computing devices. The method includes: obtaining a memory access request from a first computing device, and searching for a memory address of borrowed memory that is allocated to the first computing device and that is from a second computing device; accessing the memory address of the borrowed memory, and forwarding the memory access request to the second computing device through a target transmission link corresponding to the borrowed memory; and obtaining a processing result of the borrowed memory of the second computing device for the memory access request, and forwarding the processing result to the first computing device.

[0007] According to a second aspect, this application provides a remote memory call method, applied to a first computing device. The method includes: sending a memory access request to an intermediate device connected to a plurality of computing devices; and receiving a processing result for the memory access request, the processing result being forwarded by the intermediate device, where the processing result is obtained by borrowed memory of a second computing device processing the memory access request, the memory access request is forwarded to the second computing device by the intermediate device accessing a memory address of the borrowed memory and using a target transmission link corresponding to the borrowed memory for forwarding, and the memory address of the borrowed memory that is allocated to the first computing device and that is from the second computing device is searched for by the intermediate device after obtaining the memory access request of the first computing device.

[0008] According to a third aspect, this application provides a network interface card. The network interface card is connected to a plurality of computing devices. The network interface card includes:

a memory access request obtaining module, configured to: obtain a memory access request from a first computing device, and search for a memory address of borrowed memory that is allocated to the first computing device and that is from a second computing device;

an address access module, configured to: access the memory address of the borrowed memory, and forward the memory access request to the second computing device through a target transmission link corresponding to the borrowed memory; and

a result obtaining module, configured to: obtain a processing result of the borrowed memory of the second computing device for the memory access request, and forward the processing result to the first computing device.

[0009] According to a fourth aspect, this application provides an electronic device, including a memory, a processor, and a computer program stored in the memory. The processor

implements the method according to any one of the foregoing aspects when executing the computer program.

[0010] According to a fifth aspect, this application provides a computer-readable storage medium. The computer-readable storage medium has a computer program stored therein. When the computer program is executed by a processor, the method according to any one of the foregoing aspects is implemented.

[0011] In comparison with the related art, this application has the following advantages.

[0012] According to the embodiments of this application, a memory call mechanism between the plurality of computing devices is implemented based on the intermediate device connected to the plurality of computing devices. The intermediate device may allocate the borrowed memory that is from the second computing device for a requirement of the first computing device. Subsequently, after obtaining the memory access request from the first computing device, the intermediate device searches for the memory address of the borrowed memory of the second computing device, accesses the memory address of the borrowed memory, and forwards the memory access request to the second computing device through the target transmission link corresponding to the borrowed memory. Correspondingly, after obtaining the processing result of the borrowed memory of the second computing device for the memory access request, the intermediate device forwards the processing result to the first computing device. As can be learned, according to the foregoing solution, when the first computing device needs to call memory, especially when memory of the first computing device is insufficient, the borrowed memory can be allocated from the second computing device to the first computing device, and data processing for the first computing device can be implemented by accessing the borrowed memory of the second computing device. Through flexible allocation of memory between the computing devices, a processing capability of the first computing device is improved, which is equivalent to increasing the upper limit of memory available to the first computing device that has a borrowed-memory requirement, and the memory costs of the computing devices can be reduced. In addition, idle memory of the second computing device providing the borrowed memory is fully utilized, thereby improving resource utilization and achieving optimized scheduling of device resources.

[0013] The foregoing descriptions are merely an overview of the technical solutions in this application. To understand technical means of this application more clearly, the technical means can be implemented based on the content of the specification. In addition, to make the foregoing and other objectives, features, and advantages of this application clearer and easier to understand, specific implementations of this application are described below.

BRIEF DESCRIPTION OF THE DRAWINGS

[0014] In the accompanying drawings, unless otherwise specified, the same reference numerals throughout a plurality of accompanying drawings represent the same or similar components or elements. The accompanying drawings are not necessarily drawn to scale. It should be understood that the accompanying drawings only depict some implementations according to this application, and therefore should not be considered as limiting the scope of this application.

[0015] FIG. 1 shows a schematic diagram of an application example of a remote memory call solution according to an embodiment of this application.

[0016] FIG. 2 shows a flowchart of a remote memory call method according to an embodiment of this application.

[0017] FIG. 3 shows a flowchart of another remote memory call method according to an embodiment of this application.

[0018] FIG. 4 shows a structural block diagram of a remote memory call apparatus according to an embodiment of this application.

[0019] FIG. 5 shows a structural block diagram of another remote memory call apparatus according to an embodiment of this application.

[0020] FIG. 6 shows a structural block diagram of a network interface card according to an embodiment of this application.

[0021] FIG. 7 shows a block diagram of an electronic device configured to implement an embodiment of this application.

DETAILED DESCRIPTION

[0022] Only some exemplary embodiments are briefly described below. As a person skilled in the art may realize, the described embodiments may be modified in various different manners without departing from the idea or the scope of this application. Therefore, the accompanying drawings and the descriptions are considered to be exemplary rather than restrictive in nature.

[0023] For ease of understanding of technical solutions of the embodiments of this application, related technologies in the embodiments of this application are described below. As optional solutions, the following related technologies may be randomly combined with the technical solutions of the embodiments of this application, all of which fall within the scope of protection of the embodiments of this application.

[0024] Embodiments of this application provide a remote memory call solution. One of core ideas of the solution is: deploying a device (referred to as an intermediate device) that can be communicatively connected to a plurality of computing devices, and managing memory resources of the computing devices by using the intermediate device, so that computing devices call idle memory resources across the computing devices in a remote access manner, to supplement insufficient memory of some computing devices, and fully utilize idle memory of other computing devices. FIG. 1 is a schematic diagram of an application example used to implement a remote memory call solution according to an embodiment of this application. In the application example, an intermediate device may be a network interface card, and computing devices are respectively denoted as CN (computing node)-1 (server-1), CN-2, and CN-3. The network interface card is separately and communicatively connected to the computing devices. A MEM-HUB function is implemented using a network interface card, such as MOC (multi-function offload card). The so-called MEM-HUB is a process of implementing memory resource call across the devices through communication between the computing devices and the network interface card, with the network interface card acting as a management end for borrowing between the computing devices.

[0025] Taking an example in which CN-1 borrows memory from outside, when having insufficient memory or lacking memory of a specific type currently, CN-1 may borrow memory from other computing device(s). A memory borrowing request may be sent to the network interface card. A management interface of the network interface card determines, from CN-2 and CN-3, memory matching the memory borrowing request, and forwards the memory borrowing request to the computing device providing the borrowed memory. The computing device providing the borrowed memory marks the borrowed memory as a borrowed state. Referring to FIG. 1, for each computing device having a communication connection, the network interface card simulates a PCI device on the computing device, which is equivalent to configuring, on the computing device, a driver for communicating with the network interface card, to implement communication between the computing device and the network interface card. An implementation process of this application may be divided into a phase of allocating borrowed memory and a phase of accessing borrowed memory, which respectively correspond to a provision path (which shows the part with CN-2) and a data path shown in FIG. 1.

[0026] In the phase of allocating borrowed memory, CN-1 communicates with the network interface card through respective management interfaces. Corresponding to the provision path shown in FIG. 1, the network interface card allocates the borrowed memory from CN-2 and CN-3 based on a memory size and a memory type indicated by the memory borrowing request

of CN-1. The borrowed memory of CN-2 is high-speed memory, and is accessed through an RDMA (remote direct memory access) transmission link (transmission channel). The borrowed memory of CN-3 is low-speed memory or regular memory, and is accessed through a DMA (direct memory access) transmission link. In one case, different types of memory may need to be called based on CN-1, and therefore the different types of memory may be separately allocated from CN-2 and CN-3. In another case, the memory requirement based on CN-1 cannot be met by calling only one computing device, and parts of the memory need to be separately borrowed from CN-2 and CN-3. To enable the network interface card to obtain computing devices with memory available for borrowing, each computing device may synchronize the memory state to the network interface card side periodically or when idle memory changes, so that the network interface card determines, based on memory usage of the computing devices, the computing devices with memory available for borrowing.

[0027] The network interface card records memory addresses corresponding to the memory borrowed from CN-2 and CN-3, establishes a mapping relationship between the memory addresses and a physical address of a PCI bus register, and further notifies CN-1 of the physical address of the PCI bus register through the provision path formed by the management interfaces that respectively correspond to CN-1 and the network interface card. The physical address is an address of physical memory in CN-1 to which the PCI bus register is mapped using an MMIO (Memory-Mapped I/O, memory-mapped input/output) technology. CN-1 translates the physical address of the PCI bus register into a virtual address corresponding to a virtual machine, to subsequently initiate access to the borrowed memory based on the virtual address.

[0028] After a memory access requirement is generated based on a processing task of the virtual machine (VM) running on CN-1, an access request for the borrowed memory is sent to the network interface card. Specifically, the access request is initiated by accessing the virtual address (which is a VM/APP address space) of the PCI bus register. A memory management unit (MMU) of CN-1 maps the virtual address to a corresponding physical address, so that access to the PCI bus register can be initiated based on the physical address. The PCI bus register is equivalent to a communication channel between CN-1 and the network interface card. The memory access request for the borrowed memory may be further forwarded to the network interface card through the simulated PCI device by accessing the PCI bus register.

[0029] An address translation module of the network interface card searches an address mapping relationship for the memory addresses of the borrowed memory corresponding to the physical address of the PCI bus register, so that the memory access request can be forwarded, by accessing the memory address of the borrowed memory, to CN-2 and CN-3 that provide the

borrowed memory. Because types of the borrowed memory corresponding to CN-2 and CN-3 are different, specifically, the memory borrowing request is forwarded to CN-2 using the RDMA transmission link, and the memory borrowing request is forwarded to CN-3 through the DMA transmission link. Specifically, data sent by the RDMA transmission link is sent to CN-2 through RDMA NICs (network interface card controllers) that respectively correspond to the network interface card and CN-2. CN-2 and CN-3 obtain processing results after processing the memory access request using the borrowed memory, and further send the processing results to the network interface card. The network interface card forwards the processing results to CN-1.

[0030] After completing memory borrowing, CN-1 may send a memory deactivation request to the network interface card. The network interface card determines CN-2 and CN-3 that provide the borrowed memory to CN-1, and notifies CN-2 and CN-3 to stop lending the borrowed memory. CN-2 and CN-3 may cancel the marking of the borrowed memory.

[0031] As can be learned from the foregoing example, memory resources can be borrowed between different computing devices. When CN-1 needs to call memory, especially when memory of CN-1 is insufficient, borrowed memory can be allocated from CN-2 to CN-1, and data processing for CN-1 can be implemented by accessing the borrowed memory of CN-2. Through flexible allocation of memory between the computing devices, a processing capability of CN-1 is improved, which is equivalent to increasing the upper limit of memory available to CN-1 that has the borrowed-memory requirement, and the memory costs of CN-1 can be reduced. In addition, idle memory of CN-2 providing the borrowed memory is fully utilized, thereby improving resource utilization and achieving optimized scheduling of device resources. During one time of memory borrowing, CN-2 provides borrowed memory to CN-1. During another time of memory borrowing, CN-1 may alternatively provide borrowed memory to CN-2. From the perspective of the network interface card, symmetrical calling between the devices is implemented. The embodiments of this application are particularly suitable for processing of cold data. During data processing by a computing device, an original memory resource may be allocated for processing of hot data that is used more frequently. For cold data that is used less frequently, remote borrowed memory may be called to complete data processing when memory resources are insufficient.

[0032] The foregoing is an example of a process of remotely calling memory. During specific implementation, one network interface card may be deployed to implement one-to-many management between the network interface card and the computing devices; or a plurality of network interface cards may be deployed to implement many-to-many management

between the network interface cards and the computing devices. When the plurality of network interface cards are deployed, a borrowing request of a computing device is allocated to one of the network interface cards according to a particular rule or randomly, and this network interface card manages a borrowing process. During specific implementation, memory may be borrowed from one or more computing devices, and one or more types of borrowed memory may be provided correspondingly. In a possible implementation, the borrowed memory may alternatively be from a memory resource in the network interface card. For example, in FIG. 1, a memory resource of a MOC DDR (Double Data Rate), such as DDR SDRAM (double data rate synchronous dynamic random access memory) is called through memory management by using a memory channel. The foregoing various cases are not limited in this application.

[0033] The computing device has a function of performing memory-based operation, and may be a physical server or may be a smart terminal or other devices. The physical server may be used as a host machine on which one or more virtual machines may be deployed. The smart terminal may be a smart home device, a smart facility (for example, a smart camera), an in-vehicle device, a wearable device, a user mobile phone, a tablet computer, a notebook computer, an ultra-mobile personal computer (UMPC), a personal digital assistant (PDA), or other devices.

[0034] The intermediate device implementing memory call management between the computing devices may be a network interface card, or may be other devices or hardware, for example, a DPU (Data Processing Unit, which is a next-generation processor for data-centric computing), a CIPU (cloud infrastructure processing unit), or other hardware devices (for example, a server or a terminal device) or hardware chips (for example, a CPU, a GPU, an FPGA, an NPU, or an AI accelerator card). An entity on which memory call management is specifically implemented is not limited in this application. In an implementation, the intermediate device may be a PCI (peripheral component interconnect) device that communicates with other devices based on a PCI communication protocol. The intermediate device communicates with the computing device based on the PCI communication protocol. When the computing device is not a PCI device, communication between the intermediate device and the computing device based on the PCI communication protocol may be implemented by creating a virtual PCI device on the computing device, that is, configuring a driver for communication between the computing device and the intermediate device on the computing device.

[0035] It should be noted that various data involved in this application, for example, user information (including but not limited to user device information, user personal information,

and the like) and processed data (including but not limited to data for analysis, data for storage, data for display, and the like), is all data authorized by a user or fully authorized by each party, and collection, use, and processing of related data need to comply with relevant laws, regulations and standards of related countries and regions. In addition, a corresponding operation entry is provided for the user to choose authorization or rejection.

[0036] The following describes the technical solutions of this application and how to solve the foregoing technical problems according to the technical solutions of this application in detail by using specific embodiments. The listed several specific embodiments may be combined with each other. For the same or similar concepts or processes, details may not be described again in some embodiments. The following describes the embodiments of this application in detail with reference to the accompanying drawings.

[0037] Embodiments of this application provide a remote memory call method, applied to an intermediate device connected to a plurality of computing devices. FIG. 2 is a flowchart of a remote memory call method 200 according to an embodiment of this application. The method 200 may include the following steps.

[0038] Step S201: Obtain a memory access request from a first computing device, and search for a memory address of borrowed memory that is allocated to the first computing device and that is from a second computing device.

[0039] The first computing device requests the intermediate device to allocate the borrowed memory in advance. The intermediate device stores the memory address of the borrowed memory after allocating the borrowed memory. When the first computing device needs to access the borrowed memory, the intermediate device searches for the pre-stored memory address of the borrowed memory based on the access request for the borrowed memory, to further initiate access to the borrowed memory based on the memory address. There may be one or more second computing devices. In other words, the borrowed memory provided to the first computing device may be borrowed memory from one second computing device, or may be borrowed memory from different computing devices. For example, in FIG. 1, the second computing devices are CN-2 and CN-3. When a plurality of second computing devices are involved, memory addresses of the borrowed memory are memory addresses of the borrowed memory provided by the second computing devices, respectively.

[0040] Step S202: Access the memory address of the borrowed memory, and forward the memory access request to the second computing device through a target transmission link corresponding to the borrowed memory.

[0041] The intermediate device may further forward the memory access request of the first

computing device to the second computing device by accessing the memory address of the borrowed memory. The intermediate device accesses the borrowed memory through the target transmission link corresponding to the borrowed memory. The transmission link is a transmission channel between the devices, and correspondingly identifies an access manner in which the intermediate device accesses the borrowed memory of the second computing device. In an example, the transmission link includes a DMA transmission link or an RDMA transmission link.

[0042] In a conventional I/O data transmission process, because a CPU participates in a data transfer process and needs to occupy CPU resources, for a scenario requiring large-batch data transmission, CPU performance is greatly affected. DMA is a data direct transmission technology, and data is copied from one address space to another address space by establishing a DMA transmission link between two ends of data transmission. Specifically, when data is transmitted through the DMA transmission link, a request is sent to a DMA controller. The DMA controller obtains the transmitted data and stores the transmitted data in the DMA transmission link. A bus of the DMA controller coordinates with a bus matrix. A system bus is used to store the transmitted data at a destination position through the DMA transmission link. In comparison with the conventional data transmission manner, high-speed data transmission performance is provided, and data transfer work is entirely implemented by a DMA-related control module. The CPU only directs and controls network-interface-card-related processing, interruption processing, packet encapsulation and parsing, and the like, thereby reducing the occupation of a large part of CPU resources. As a remote direct access technology, RDMA can transmit data from one device to another device, and is dedicated to resolving a problem of processing latency during data transmission between the devices. CPUs of the devices at two ends only participate in the control plane, and hardly participate in a data transmission process. During the transmission process, the network interface card directly copies data from the user space of memory to an internal storage space via DMA, and then hardware assembles various layers of packets, and sends an assembled packet to a peer network interface card through a physical link. After receiving the data, an RDMA network interface card at the receiving end strips off packet headers and checksums of the layers, and directly copies the data to the user space memory via DMA, thereby achieving both higher bandwidth and lower latency. Data transmission is performed in combination with high-speed memory, so that the efficiency of data transmission can be greatly improved.

[0043] In a phase of allocating borrowed memory, after allocating the borrowed memory, the second computing device may perform a handshake with the intermediate device in a

protocol communication manner to provide a specific access method for memory access. Taking an RDMA communication link as an example, information such as an IP address and a physical port address of the second computing device needs to be provided to the intermediate device in a handshake phase, so that when data is subsequently transmitted through the RDMA communication link, data transmission is completed based on the information provided during the handshake. The IP address and the physical port address may be stored in the address mapping relationship, so that data transmission is performed directly based on the IP address and the physical port address in a process of performing data transmission. Taking a DMA communication link as an example, information such as PCI port information is provided to the intermediate device in a handshake phase. Correspondingly, the information such as the PCI port information may be provided to the intermediate device to be stored in the address mapping relationship.

[0044] A correspondence relationship between the borrowed memory and the target transmission link may be configured according to a specific requirement. This is not limited in this application. For example, a corresponding transmission link is configured based on a memory type of the borrowed memory.

[0045] Step S203: Obtain a processing result of the borrowed memory of the second computing device for the memory access request, and forward the processing result to the first computing device.

[0046] The second computing device calls the borrowed memory to process the memory access request forwarded by the intermediate device, and the processing result will be sent to the intermediate device. The intermediate device further sends the processing result to the first computing device.

[0047] As can be learned, according to the foregoing solution, when the first computing device needs to call memory, especially when memory of the first computing device is insufficient, the borrowed memory can be allocated from the second computing device to the first computing device, and data processing for the first computing device can be implemented by accessing the borrowed memory of the second computing device. Through flexible allocation of memory between the computing devices, a processing capability of the first computing device is improved, which is equivalent to increasing the upper limit of memory available to the first computing device that has a borrowed-memory requirement, and the memory costs of the computing devices can be reduced. In addition, idle memory of the second computing device providing the borrowed memory is fully utilized, thereby improving resource utilization and achieving optimized scheduling of device resources.

[0048] Before obtaining the memory access request from the first computing device, the borrowed memory needs to be first allocated to the first computing device. The first computing device may send a memory borrowing request to the intermediate device as required. The intermediate device obtains the memory borrowing request sent by the first computing device, determines the second computing device matching the memory borrowing request from at least one other computing device, and forwards the memory borrowing request to the second computing device. The memory borrowing request may carry a required memory type and a size of the borrowed memory. The intermediate device further determines, based on the memory type and the size of the borrowed memory, computing devices from which memory is borrowed. For example, if the memory type is high-speed memory and low-speed memory, a computing device having high-speed memory and low-speed memory is determined as the second computing device, or computing devices that respectively have high-speed memory and low-speed memory are both determined as the second computing devices, for example, CN-2 and CN-3 shown in FIG. 1. If it is determined based on the size of the memory that a computing device can meet a requirement, the computing device is determined as the second computing device providing the borrowed memory. If one computing device is insufficient to meet the size of the required memory, the borrowed memory may be obtained from a plurality of computing devices, and the plurality of computing devices are determined as the second computing devices. After allocating the borrowed memory based on the memory borrowing request of the first computing device, the second computing device may notify the intermediate device of the memory address of the borrowed memory, for corresponding borrowed memory to be accessed based on the memory address in a memory access phase.

[0049] When the intermediate device and the computing device are both PCI devices, the two PCI devices may communicate according to a PCI protocol. When the intermediate device is a PCI device but the computing device is not a PCI device, in a possible implementation, for each computing device, the intermediate device simulates one PCI device on the computing device, that is, a driver is deployed to enable the computing device to communicate with the intermediate device in a mode of the PCI device. In this case, the first computing device may access the intermediate device by accessing PCI bus registers, which is equivalent to interaction between two PCI devices. Data exchange between the PCI devices is completed by accessing physical addresses of the PCI bus registers. After a configuration space of the PCI device is initialized, the PCI device has an independent PCI bus address space, which is stored in the bus registers, in a current PCI bus tree. In an example, there are six registers in the configuration space of the PCI device. Each register corresponds to physical addresses of a group of PCI bus

registers used by the PCI device. When a plurality of virtual machines are deployed on the computing device, each virtual machine may communicate with other PCI devices via physical addresses of different PCI bus registers.

[0050] In an implementation, the physical address of the PCI bus register is an address of physical memory in CN-1 to which the PCI bus register is mapped using an MMIO technology, so that access to the PCI bus register is realized by accessing the physical address of the physical memory. In comparison with a conventional register access manner, advantages such as fast access speed and high flexibility are provided.

[0051] In the embodiments of this application, in the process of obtaining the memory access request from the first computing device, the intermediate device obtains the memory access request that is initiated by accessing the PCI bus register by the first computing device. The memory access request carries the physical address of the PCI bus register. Correspondingly, when searching for the memory address of the borrowed memory that is allocated to the first computing device and that is from a second computing device, the intermediate device may search an address mapping relationship corresponding to the borrowed memory for the memory address of the borrowed memory that corresponds to the physical address of the PCI bus register carried in the memory access request, so that the borrowed memory may be accessed based on the memory address of the borrowed memory.

[0052] The address mapping relationship stores a mapping relationship between the physical address of the PCI bus register and the memory address of the borrowed memory. This correspondence relationship may be created after the borrowed memory of the second computing device is allocated to the first computing device. Specifically, in the phase of allocating borrowed memory, the memory borrowing request sent by the first computing device to the intermediate device carries the physical address of the PCI bus register. After obtaining the memory address of the borrowed memory allocated by the second computing device for the memory borrowing request of the first computing device, the intermediate device establishes the address mapping relationship between the memory address of the borrowed memory and the physical address of the PCI bus register carried in the memory borrowing request, so that in the phase of accessing borrowed memory, after obtaining the physical address of the PCI bus register carried in the memory borrowing request of the first computing device, the intermediate device determines the memory address of the borrowed memory based on the address mapping relationship.

[0053] In the phase of allocating borrowed memory, after the address mapping relationship is established, a notification message indicating a borrowing success may be further sent to the

first computing device. The notification message may carry the physical address of the PCI bus register, so that the first computing device learns that the borrowed memory is successfully allocated for the memory borrowing request initiated by the PCI bus register, and can subsequently access the intermediate device by accessing the physical address of the PCI bus register, thereby further implementing access to the borrowed memory.

[0054] The following provides an example of the address mapping relationship.

First computing device ID	Physical address of bus register	Second computing device ID	Memory address of borrowed memory	Memory type
1	0×300000 - 0×700000	2	0×100000 - 0×500000	DMA
1	0×0 - 0×100000	3	0×800000 - 0×900000	RDMA
1	0×500000 - 0×600000	N/A	0×600000 - 0×700000	MOC MEM

Table 1

[0055] In this example, the address mapping relationship is stored in a data table, and the physical address of the PCI bus register and the memory address of the borrowed memory are both represented in the form of an address range. The memory address corresponding to the physical address of the PCI register bus can be determined through table look-up. For example, in Table 1, in the second computing device whose ID is 2, a memory address within the address range of 0×100000 to 0×500000 corresponds to a physical address of a register bus within the address range of 0×300000 to 0×700000 in the first computing device whose ID is 1. If the memory access request carries the physical address of the register bus within the address range of 0×300000 to 0×700000, it can be determined that an address range of a corresponding memory address is 0×100000 to 0×500000.

[0056] The physical address of the PCI bus register and the memory address of the borrowed memory are further associated with a device identifier (for example, a device ID) of a corresponding computing device. In other words, when the borrowed memory is accessed, the address mapping relationship is searched for the corresponding second computing device ID and the corresponding memory address of the borrowed memory based on the first computing device ID and the physical address of the bus register, and access to the borrowed memory can be performed based on the found second computing device ID and the found memory address of the borrowed memory. Table 1 above provides a correspondence relationship between the first computing device ID, the physical address of the bus register, the second computing device ID, and the memory address of the borrowed memory.

[0057] During specific implementation, one or more virtual machines may be deployed on

the first computing device. When a plurality of virtual machines are deployed, based on the memory borrowing requests and the memory access requests generated by different virtual-machines running tasks, interaction with the intermediate device may be performed through different PCI bus registers. The intermediate device does not sense which virtual machine an interaction process originates from. A difference lies only in that the memory borrowing requests and the memory access requests sent based on physical addresses of the different PCI bus registers are received. Correspondingly, when the address mapping relationship is established, because a correspondence relationship exists between the physical address of the PCI bus register and the memory address of the borrowed memory, borrowed memory corresponding to different virtual machines corresponds to different PCI bus registers. Referring to the computing devices in the first column of the foregoing table, whose first computing device IDs are all numbered as 1, three physical addresses of corresponding bus registers are listed in the second column, which represents that three virtual machines respectively corresponding to the computing device whose number is 1 correspond to three different PCI bus registers, that is, the three virtual machines separately interact with the intermediate device through the different PCI bus registers. The table above is merely an example. During specific implementation, the table may include a plurality of first computing devices or second computing devices. A specific form of a computing device ID and a representation form of the address range may be selected according to an actual requirement. This is not limited in this application.

[0058] A correspondence relationship between the virtual machine and the PCI bus register is determined in the phase of allocating borrowed memory. The notification message which indicates a borrowing success and which is sent to the first computing device by the intermediate device after establishing the address mapping relationship may carry the physical address of the PCI bus register, and the first computing device may establish the correspondence relationship between the virtual machine and the PCI bus register based on the physical address of the PCI bus register. Specifically, to avoid a potential security risk brought by direct access to the physical address by the virtual machine, the virtual machine of the first computing device may access the PCI bus register in a standard memory mapping manner. In other words, the physical address of the PCI bus register fed back by the intermediate device may be mapped to a virtual address (GPA, that is, guest physical address) in the virtual machine, and then the virtual address is provided to the virtual machine for subsequent access. Because the physical address is invisible to the virtual machine, remote calling at a bottom layer may be hidden from the virtual machine, and the whole process of remotely calling memory in

upper-layer processing does not need to be managed, thereby avoiding various data security problems.

[0059] In a possible implementation, when the memory address of the borrowed memory is accessed and the memory access request is forwarded to the second computing device through the target transmission link corresponding to the borrowed memory, the target transmission link used in a current memory borrowing process may be first determined. Specifically, a type of the target transmission link is determined based on the memory type of the borrowed memory. As described above, the memory type may be classified into high-speed memory, low-speed memory, and the like. Correspondingly, when a corresponding transmission link is configured based on the type of the borrowed memory, in an example, a faster transmission link may be allocated to high-speed memory, and low-speed memory or regular memory may be accessed through another transmission link. For example, if the borrowed memory is high-speed memory, memory access may be performed through the RDMA transmission link; if the borrowed memory is low-speed memory or regular memory, memory access may be performed through the DMA transmission link. In another example, to improve the access speed of low-speed memory, if the borrowed memory is low-speed memory or regular memory, the RDMA transmission link is preferentially used for memory access, and for high-speed memory, the DMA transmission link is configured for memory access. After the target transmission link is determined, the memory access request may be forwarded to the second computing device by accessing the memory address of the borrowed memory and using the determined target transmission link. Here, RDMA and DMA are merely examples, and other suitable transmission links may be used. This is not limited in this application. The memory type of the borrowed memory may be stored in the address mapping relationship after the borrowed memory is allocated, so that the memory type can be directly read from the address mapping relationship, and the corresponding target transmission link can be determined based on the memory type. As shown in Table 1 above, the memory type is recorded in the address mapping relationship, and the memory type of the borrowed memory may be determined via the physical address of the PCI bus register (which may be further combined with the first computing device ID) carried in the memory access request.

[0060] Correspondingly, when forwarding the processing result of the borrowed memory of the second computing device for the memory access request to the first computing device after obtaining the processing result, the intermediate device may obtain the processing result that is returned by the second computing device for the memory access request through the determined target transmission link, for example, the processing result is transmitted to the

intermediate device in an RDMA manner or a DMA manner. The intermediate device then forwards the processing result to the first computing device through the PCI bus register. Specifically, when the intermediate device accesses the PCI bus register of the first computing device via the physical address, a PCI bus controller converts the processing result into a PCI protocol data packet, and then the processing result in the form of the PCI protocol data packet is sent to the virtual machine that uses the borrowed memory through a PCI bus.

[0061] After the memory borrowing ends, the first computing device may send a memory deactivation request to the intermediate device. After obtaining the memory deactivation request from the first computing device, the intermediate device may search for the second computing device that provides the borrowed memory to the first computing device, and notify the second computing device to stop lending the borrowed memory. In this case, the borrowing process of the borrowed memory is completed. After the borrowed memory is allocated to the first computing device, the second computing device providing the borrowed memory for the first computing device may be recorded. In a possible implementation, a correspondence relationship between the first computing device and the second computing device may be recorded in the above address mapping relationship for storing the physical address of the PCI bus register and the memory address. When a plurality of virtual machines are deployed on the first computing device, the respective borrowed memory corresponding to the virtual machines on the first computing device may be correspondingly from different second computing devices. The device identifier of the corresponding second computing device may be determined with reference to the physical address of the PCI bus register and the device identifier of the first computing device that are carried when the memory deactivation request is initiated, so as to notify the second computing device to stop lending the borrowed memory. The memory address of the borrowed memory may be further provided to the second computing device for the second computing device to stop lending the borrowed memory based on the memory address.

[0062] After the borrowing of the borrowed memory is stopped, the corresponding data recorded for the borrowed memory may be further deleted from the address mapping relationship recorded on the intermediate device.

[0063] Embodiments of this application further provide a remote memory call method, applied to the foregoing first computing device. FIG. 3 is a flowchart of a remote memory call method 300 according to an embodiment of this application. The method 300 may include the following steps.

[0064] Step S301: Send a memory access request to an intermediate device connected to a plurality of computing devices.

[0065] Step S302: Receive a processing result for the memory access request, the processing result being forwarded by the intermediate device. Herein, the processing result is obtained by borrowed memory of a second computing device processing the memory access request. The memory access request is forwarded to the second computing device by the intermediate device accessing a memory address of the borrowed memory and using a target transmission link corresponding to the borrowed memory for forwarding. The memory address of the borrowed memory that is allocated to the first computing device and that is from the second computing device is searched for by the intermediate device after obtaining the memory access request of the first computing device.

[0066] This embodiment describes a process of implementing the solution from the perspective of the first computing device. When needing to access the borrowed memory, the first computing device sends the memory access request to the intermediate device, and the intermediate device forwards the memory access request to one or more second computing devices that provide the borrowed memory. The second computing devices call the borrowed memory to process the memory access request to obtain the processing results. The processing results are forwarded by the intermediate device to the first computing device. For a part of the content, please refer to the description of the previous embodiment. Details are not described herein again.

[0067] In the process of sending the memory access request to the intermediate device connected to the plurality of computing devices, the first computing device sends the memory access request to the intermediate device by accessing a PCI bus register. The memory access request carries a physical address of the PCI bus register for the intermediate device to search an address mapping relationship corresponding to borrowed memory for the memory address of the borrowed memory that corresponds to the physical address of the PCI bus register carried in the memory access request.

[0068] The memory access request may be generated by a virtual machine based on a processing task. Based on the description of the previous embodiment, access to the PCI bus register is initiated by accessing the physical address of the PCI bus register, and then the memory access request is sent to the intermediate device. The memory access request carries the physical address of the PCI bus register. The intermediate device may further search the address mapping relationship corresponding to the borrowed memory for the memory address of the borrowed memory corresponding to the physical address of the PCI bus register carried in the memory access request.

[0069] In a possible implementation of the embodiments of this application, the virtual

machine may access the PCI bus register in a standard memory mapping manner. In other words, the physical address is mapped to a virtual address through a memory mapping mechanism, so that the PCI bus register is accessed via the virtual address. Specifically, the virtual machine initiates the memory access request based on the virtual address of the PCI bus register in the virtual machine. A memory management unit of the first computing device may map the virtual address corresponding to the PCI bus register in the virtual machine to a corresponding physical address in a physical machine, and the PCI bus register is accessed via the physical address. A PCI bus controller (Root Complex) of the first computing device is an interface between a system CPU and a PCI bus, and acts on behalf of the system CPU to communicate with other components. The PCI bus controller converts the memory access request into a PCI protocol data packet based on access to the PCI bus register, and sends the PCI protocol data packet to the intermediate device through the PCI bus. The intermediate device receives the memory access request in the form of the PCI protocol data packet, determines that the memory access request is from a simulated PCI device, and performs, based on the memory access request, a next step of accessing the borrowed memory. Remote calling at a bottom layer can be hidden from the virtual machine by mapping the physical address of the PCI bus register to the virtual address and providing the virtual address to the virtual machine for access, and the whole process of remotely calling memory in upper-layer processing does not require management, thereby avoiding various data security problems.

[0070] During specific implementation of standard memory mapping, address mapping from the perspective of the virtual machine may be implemented using an EPT (extended page table), or address mapping for the PCI device may be implemented using an IOPT (I/O MMU Page Table, I/O memory management unit page table), or address mapping in a host mode may be implemented using a PT (page table), or other manners may be used. This is not limited in this application.

[0071] In a possible implementation, before the memory access request is sent to the intermediate device connected to the plurality of computing devices, it is also possible to translate the physical address into the virtual address after the borrowed memory is allocated by the intermediate device, and bind the virtual address to the virtual machine initiating a memory borrowing request, so that the virtual machine can initiate the memory access request by accessing the PCI bus register via the virtual address. In a possible implementation, before the memory access request is sent to the intermediate device connected to the plurality of computing devices, it is also possible to obtain the physical address of the PCI bus register sent by the intermediate device after the borrowed memory is allocated, translate the physical

address into the virtual address, and bind the virtual address to the virtual machine initiating a memory borrowing request, so that the virtual machine can initiate the memory access request by accessing the PCI bus register via the virtual address. During specific implementation, the binding the virtual address of the PCI bus register to the virtual machine initiating the memory borrowing request may be directly mapping the virtual addresses range of the PCI bus register to a memory space of the virtual machine in a manner of a page table.

[0072] Before the process of accessing the borrowed memory is performed, it is further necessary to request allocation of borrowed memory from other computing device(s) through the intermediate device. Therefore, before sending the memory access request to the intermediate device connected to the plurality of computing devices, the memory borrowing request may be further sent to the intermediate device, so as to allocate the borrowed memory matching the memory borrowing request from the second computing device(s) through the intermediate device.

[0073] When it is determined that it is not necessary to continue to call the allocated borrowed memory from other computing device(s), a memory deactivation request for the borrowed memory may be further sent to the intermediate device, so that the intermediate device notifies the second computing device(s) to stop lending the borrowed memory.

[0074] During specific implementation, the computing device may report a memory usage status to the intermediate device periodically or when idle memory changes, so that the intermediate device may monitor the memory status of the computing device, and further determine which computing devices the memory borrowing request is to be forwarded to when it is necessary to request the borrowed memory.

[0075] In the foregoing embodiments, the first computing device serves as a requester of the borrowed memory to request the borrowed memory from the second computing device through the intermediate device. During specific implementation, the first computing device may also serve as a provider of the borrowed memory to provide an idle memory resource of the first computing device to other computing device(s). For example, when a third computing device requires borrowed memory, the third computing device sends a memory borrowing request to the intermediate device. The intermediate device determines, based on the memory status of each computing device, that the first computing device may serve as the provider of the borrowed memory and may forward the memory borrowing request to the first computing device. After receiving the memory borrowing request of the third computing device sent by the intermediate device, the first computing device allocates the idle memory and adds an identifier of the borrowed memory.

[0076] Further, a memory type of the borrowed memory and a memory-related parameter may be returned to the intermediate device. The memory type may be used for further determining a memory borrowing manner, that is, which kind of transmission link is used to access the borrowed memory. For example, if the memory type is high-speed memory, memory access may be performed through an RDMA transmission link. The memory-related parameter is a memory address. For example, low-speed memory provides memory access through a DMA transmission link, and the memory address is a physical address of the memory. The high-speed memory provides memory access through the RDMA transmission link, and the memory address is a device port number and a physical address of the memory.

[0077] Corresponding to the method embodiments provided in the embodiments of this application, embodiments of this application further provide a remote memory call apparatus, applied to an intermediate device connected to a plurality of computing devices. FIG. 4 is a structural block diagram of a remote memory call apparatus 400 according to an embodiment of this application. The apparatus 400 may include:

an access request obtaining module 401, configured to obtain a memory access request from a first computing device;

a memory address search module 402, configured to search for a memory address of borrowed memory that is allocated to the first computing device and that is from a second computing device;

an access request forwarding module 403, configured to: access the memory address of the borrowed memory, and forward the memory access request to the second computing device through a target transmission link corresponding to the borrowed memory; and

a result forwarding module 404, configured to: obtain a processing result of the borrowed memory of the second computing device for the memory access request, and forward the processing result to the first computing device.

[0078] In a possible implementation, the apparatus further includes:

a borrowing request obtaining module, configured to obtain, before the access request obtaining module 401 obtains the memory access request from the first computing device, a memory borrowing request sent by the first computing device, where the memory borrowing request indicates a size of the borrowed memory and a memory type of the borrowed memory;

a borrowing request forwarding module, configured to: determine the second computing device matching the memory borrowing request from at least one other computing

device, and forward the memory borrowing request to the second computing device; and

a memory address obtaining module, configured to obtain the memory address of the borrowed memory allocated by the second computing device.

[0079] In a possible implementation, the access request obtaining module is specifically configured to obtain the memory access request that the first computing device initiates by accessing a PCI bus register, where the memory access request carries a physical address of the PCI bus register. Correspondingly, the memory address search module is specifically configured to search an address mapping relationship corresponding to the borrowed memory for the memory address of the borrowed memory that corresponds to the physical address of the PCI bus register carried in the memory access request, where the address mapping relationship is created by the intermediate device after allocating the borrowed memory.

[0080] In a possible implementation, the apparatus further includes:

a mapping relationship establishment module, configured to establish the address mapping relationship between the memory address of the borrowed memory and the physical address of the PCI bus register after obtaining the memory address of the borrowed memory that is allocated by the second computing device for the memory borrowing request of the first computing device and before the access request obtaining module 401 obtains the memory access request from the first computing device; and

a physical address notification module, configured to notify the first computing device of the physical address of the PCI bus register.

[0081] In a possible implementation, the access request forwarding module includes:

a transmission link determining submodule, configured to determine the target transmission link used in a current memory borrowing process, where a type of the target transmission link is determined based on the memory type of the borrowed memory and/or service level information corresponding to the memory access request; and

a memory address access submodule, configured to: access the memory address of the borrowed memory, and forward the memory access request to the second computing device through the determined target transmission link.

[0082] In a possible implementation, the result forwarding module is specifically configured to: obtain the processing result that is returned by the second computing device for the memory access request through the target transmission link, and forward the processing result to the first computing device through the PCI bus register.

[0083] In a possible implementation, the apparatus further includes:

a deactivation request obtaining module, configured to obtain a memory

deactivation request from the first computing device; and

a borrowing stop module, configured to: search for the second computing device that provides the borrowed memory to the first computing device, and notify the second computing device to stop lending the borrowed memory.

[0084] According to the embodiments of this application, a memory call mechanism between the plurality of computing devices is implemented based on the intermediate device connected to the plurality of computing devices. The intermediate device may allocate the borrowed memory that is from the second computing device for a requirement of the first computing device. Subsequently, after obtaining the memory access request from the first computing device, the intermediate device searches for the memory address of the borrowed memory of the second computing device, accesses the memory address of the borrowed memory, and forwards the memory access request to the second computing device through the target transmission link corresponding to the borrowed memory. Correspondingly, after obtaining the processing result of the borrowed memory of the second computing device for the memory access request, the intermediate device forwards the processing result to the first computing device. As can be learned, according to the foregoing solution, when the first computing device needs to call memory, especially when memory of the first computing device is insufficient, the borrowed memory can be allocated from the second computing device to the first computing device, and data processing for the first computing device can be implemented by accessing the borrowed memory of the second computing device. Through flexible allocation of memory between the computing devices, a processing capability of the first computing device is improved, which is equivalent to increasing the upper limit of memory available to the first computing device that has a borrowed-memory requirement, and the memory costs of the computing devices can be reduced. In addition, idle memory of the second computing device providing the borrowed memory is fully utilized, thereby improving resource utilization and achieving optimized scheduling of device resources.

[0085] Corresponding to the method embodiments provided in the embodiments of this application, embodiments of this application further provide a remote memory call apparatus, applied to a first computing device. FIG. 5 is a structural block diagram of a remote memory call apparatus 500 according to an embodiment of this application. The apparatus 500 may include:

an access request sending module 501, configured to send a memory access request to an intermediate device connected to a plurality of computing devices; and

a processing result receiving module 502, configured to receive a processing result

for the memory access request, the processing result being forwarded by the intermediate device, where the processing result is obtained by borrowed memory of a second computing device processing the memory access request, the memory access request is forwarded to the second computing device by the intermediate device accessing a memory address of the borrowed memory and using a target transmission link corresponding to the borrowed memory for forwarding, and the memory address of the borrowed memory that is allocated to the first computing device and that is from the second computing device is searched for by the intermediate device after obtaining the memory access request of the first computing device.

[0086] In a possible implementation, the access request sending module is specifically configured to send the memory access request to the intermediate device by accessing a PCI bus register, where the memory access request carries a physical address of the PCI bus register, for the intermediate device to search an address mapping relationship corresponding to borrowed memory for the memory address of the borrowed memory that corresponds to the physical address of the PCI bus register carried in the memory access request.

[0087] In a possible implementation, the access request sending module includes:

an access request initiation submodule, configured to initiate the memory access request based on a virtual address of the PCI bus register in a virtual machine of the first computing device;

an address translation submodule, configured to translate the virtual address of the PCI bus register into a physical address in the first computing device; and

a request sending submodule, configured to: access the PCI bus register via the physical address in the first computing device, and send the memory access request to the intermediate device.

[0088] In a possible implementation, the apparatus further includes:

an address processing module, configured to: before the access request sending module 501 sends the memory access request to the intermediate device connected to the plurality of computing devices, translate the physical address of the PCI bus register into the virtual address after the borrowed memory is allocated by the intermediate device, and bind the virtual address to the virtual machine initiating a memory borrowing request, for the virtual machine to initiate the memory access request by accessing the PCI bus register via the virtual address.

[0089] In a possible implementation, the address processing module is specifically configured to :

before the access request sending module 501 sends the memory access request to

the intermediate device connected to the plurality of computing devices, obtain the physical address of the PCI bus register that is sent by the intermediate device after the borrowed memory is allocated, translate the physical address into the virtual address, and bind the virtual address to the virtual machine initiating a memory borrowing request, for the virtual machine to initiate the memory access request by accessing the PCI bus register via the virtual address.

[0090] In a possible implementation, the apparatus further includes:

a borrowing request sending module, configured to: before the access request sending module 501 sends the memory access request to the intermediate device connected to the plurality of computing devices, send the memory borrowing request to the intermediate device, to allocate the borrowed memory matching the memory borrowing request from the second computing device through the intermediate device.

[0091] In a possible implementation, the apparatus further includes:

a deactivation request sending module, configured to send a memory deactivation request for the borrowed memory to the intermediate device, so that the intermediate device notifies the second computing device to stop lending the borrowed memory.

[0092] In a possible implementation, the apparatus further includes:

a status reporting module, configured to report a memory usage status to the intermediate device;

a request receiving module, configured to receive a memory borrowing request of a third computing device sent by the intermediate device;

a memory marking module, configured to: allocate idle memory from the first computing device, and add an identifier of the borrowed memory.

[0093] According to the embodiments of this application, a memory call mechanism between the plurality of computing devices is implemented based on the intermediate device connected to the plurality of computing devices. The intermediate device may allocate the borrowed memory that is from the second computing device for a requirement of the first computing device. Subsequently, after obtaining the memory access request from the first computing device, the intermediate device searches for the memory address of the borrowed memory of the second computing device, accesses the memory address of the borrowed memory, and forwards the memory access request to the second computing device through the target transmission link corresponding to the borrowed memory. Correspondingly, after obtaining the processing result of the borrowed memory of the second computing device for the memory access request, the intermediate device forwards the processing result to the first computing device. As can be learned, according to the foregoing solution, when the first

computing device needs to call memory, especially when memory of the first computing device is insufficient, the borrowed memory can be allocated from the second computing device to the first computing device, and data processing for the first computing device can be implemented by accessing the borrowed memory of the second computing device. Through flexible allocation of memory between the computing devices, a processing capability of the first computing device is improved, which is equivalent to increasing the upper limit of memory available to the first computing device that has a borrowed-memory requirement, and the memory costs of the computing devices can be reduced. In addition, idle memory of the second computing device providing the borrowed memory is fully utilized, thereby improving resource utilization and achieving optimized scheduling of device resources.

[0094] Corresponding to the method embodiments and the apparatus embodiments provided in the embodiments of this application, embodiments of this application further provide a network interface card. The network interface card is connected to a plurality of computing devices. FIG. 6 is a structural block diagram of a network interface card 600 according to an embodiment of this application. The network interface card 600 may include:

a memory access request obtaining module 601, configured to: obtain a memory access request from a first computing device, and search for a memory address of borrowed memory that is allocated to the first computing device and that is from a second computing device;

an address access module 602, configured to: access the memory address of the borrowed memory, and forward the memory access request to the second computing device through a target transmission link corresponding to the borrowed memory; and

a result obtaining module 603, configured to: obtain a processing result of the borrowed memory of the second computing device for the memory access request, and forward the processing result to the first computing device.

[0095] According to the embodiments of this application, a memory call mechanism between the plurality of computing devices is implemented based on the intermediate device connected to the plurality of computing devices. The intermediate device may allocate the borrowed memory that is from the second computing device for a requirement of the first computing device. Subsequently, after obtaining the memory access request from the first computing device, the intermediate device searches for the memory address of the borrowed memory of the second computing device, accesses the memory address of the borrowed memory, and forwards the memory access request to the second computing device through the target transmission link corresponding to the borrowed memory. Correspondingly, after

obtaining the processing result of the borrowed memory of the second computing device for the memory access request, the intermediate device forwards the processing result to the first computing device. As can be learned, according to the foregoing solution, when the first computing device needs to call memory, especially when memory of the first computing device is insufficient, the borrowed memory can be allocated from the second computing device to the first computing device, and data processing for the first computing device can be implemented by accessing the borrowed memory of the second computing device. Through flexible allocation of memory between the computing devices, a processing capability of the first computing device is improved, which is equivalent to increasing the upper limit of memory available to the first computing device that has a borrowed-memory requirement, and the memory costs of the computing devices can be reduced. In addition, idle memory of the second computing device providing the borrowed memory is fully utilized, thereby improving resource utilization and achieving optimized scheduling of device resources.

[0096] The functions of the modules in the apparatuses in the embodiments of this application can be found in the corresponding descriptions in the foregoing methods, and have corresponding beneficial effects. Details are not described herein again.

[0097] FIG. 7 is a block diagram of an electronic device configured to implement an embodiment of this application. As shown in FIG. 7, the electronic device includes a memory 701 and a processor 702. The memory 701 stores a computer program executable on the processor 702. The processor 702, when executing the computer program, implements the method according to the foregoing embodiments. One or more memories 701 and processors 702 may be provided.

[0098] The electronic device further includes:

a communication interface 703, configured to communicate with an external device to perform data exchange and transmission.

[0099] If the memory 701, the processor 702, and the communication interface 703 are independently implemented, the memory 701, the processor 702, and the communication interface 703 may be connected to each other by a bus and complete mutual communication. The bus may be an industry standard architecture (ISA) bus, a peripheral component interconnect (PCI) bus, an extended industry standard architecture (EISA) bus, or the like. The bus may be classified into an address bus, a data bus, a control bus, and the like. For ease of representation, only one thick line is used to represent the bus in FIG. 7, but this does not mean that only one bus or only one type of bus is provided.

[0100] Optionally, during specific implementation, if the memory 701, the processor 702,

and the communication interface 703 are integrated on a chip, the memory 701, the processor 702, and the communication interface 703 may complete mutual communication through an internal interface.

[0101] Embodiments of this application provide a computer-readable storage medium. The computer-readable storage medium has a computer program stored therein. When the program is executed by a processor, the method according to the embodiments of this application is implemented.

[0102] Embodiments of this application further provide a computer program product. The computer program product includes a computer program. When the computer program is executed by a processor, the method according to any embodiment of this application is implemented.

[0103] Embodiments of this application further provide a chip. The chip includes a processor, configured to call instructions stored in a memory from the memory and run the instructions, so that a communication device on which the chip is installed performs the method according to the embodiments of this application.

[0104] Embodiments of this application further provide a chip, including an input interface, an output interface, a processor, and a memory. The input interface, the output interface, the processor, and the memory are connected through an internal connection path. The processor is configured to execute code in the memory, and when the code is executed, the processor is configured to perform the method according to the embodiments of this application.

[0105] It should be understood that the foregoing processor may be a central processing unit (CPU), or may be another general-purpose processor, a digital signal processor (DSP), an application specific integrated circuit (ASIC), a field programmable gate array (FPGA), another programmable logic device, a discrete gate or transistor logic device, a discrete hardware component, or the like. The general-purpose processor may be a microprocessor, any regular processor, or the like. It should be noted that the processor may be a processor that supports an advanced RISC machine (Advanced Reduced Instruction Set Computer Machine, ARM) architecture.

[0106] Further, optionally, the foregoing memory may include a read-only memory and a random access memory. The memory may be a volatile memory or a non-volatile memory, or may include both a volatile memory and a non-volatile memory. The non-volatile memory may include a read-only memory (ROM), a programmable read-only memory (PROM), an erasable programmable read-only memory (EPROM), an electrically erasable programmable read-only memory (EEPROM), or a flash memory. The volatile memory may include a random access

memory (RAM), which serves as an external cache. Through exemplary but not restrictive descriptions, many forms of RAMs are available, for example, a static random access memory (SRAM), a dynamic random access memory (DRAM), a synchronous dynamic random access memory (SDRAM), a double data rate synchronous dynamic random access memory (DDR SDRAM), an enhanced synchronous dynamic random access memory (ESDRAM), a synchronous link dynamic random access memory (Sync link DRAM, SLDRAM), and a direct Rambus random access memory (DR RAM).

[0107] All or some of the foregoing embodiments may be implemented through software, hardware, firmware, or any combination thereof. When the software is used for implementation, all or some of the foregoing embodiments may be implemented in the form of a computer program product. The computer program product includes one or more computer instructions. When the computer program instructions are loaded and executed on a computer, the procedures or functions according to this application are all or partially generated. The computer may be a general-purpose computer, a special-purpose computer, a computer network, or another programmable apparatus. The computer instructions may be stored in a computer-readable storage medium, or may be transmitted from one computer-readable storage medium to another computer-readable storage medium.

[0108] In the descriptions of the specification, the descriptions of reference terms such as "a/an/one embodiment", "some embodiments", "example", "specific example", or "some examples" mean that specific features, structures, materials or characteristics described with reference to the embodiment or example are included in at least one embodiment or example of this application. In addition, the described specific features, structures, materials, or characteristics may be combined in an appropriate manner in any one or more embodiments or examples. In addition, a person skilled in the art may integrate or combine different embodiments or examples described in the specification and features of the different embodiments or examples as long as they are not contradictory to each other.

[0109] In addition, terms "first" and "second" are used merely for the purpose of description, and cannot be construed as indicating or implying relative importance or implying a quantity of indicated technical features. Therefore, a feature defined by "first" or "second" can explicitly or implicitly include at least one such feature. In the descriptions of this application, "a plurality of" means two or more, unless otherwise definitely and specifically limited.

[0110] Any process or method described in the flowcharts or in other manners herein may be understood as representing a module, a segment, or a part of code including one or more

executable instructions for implementing steps of a specific logical function or process. In addition, a range of preferred implementations of this application includes additional implementations, in which functions may be performed in a manner that is not necessarily in the order shown or discussed, including in a substantially simultaneous manner or in a reverse order, depending on the functions involved.

[0111] The logic and/or the steps described in the flowcharts or in other manners herein, for example, an ordered listing that may be considered as executable instructions for implementing logical functions, may be specifically implemented in any computer-readable medium to be used by instruction execution systems, apparatuses, or devices (for example, a computer-based system, a system including a processor, or other systems that can obtain instructions from the instruction execution systems, apparatuses, or devices and execute the instructions) or to be used in combination with such instruction execution systems, apparatuses, or devices.

[0112] It should be understood that the parts of this application can be implemented by using hardware, software, firmware, or a combination thereof. In the foregoing implementations, a plurality of steps or methods may be implemented by using software or firmware that is stored in a memory and executed by an appropriate instruction execution system. All or a part of the steps of the method in the foregoing embodiments may be implemented by a program instructing relevant hardware. The program may be stored in a computer-readable storage medium. When the program is executed, one or a combination of the steps of the method embodiments are included.

[0113] In addition, functional units in the embodiments of this application may be integrated into one processing module, or each of the units may exist alone physically, or two or more units may be integrated into one module. The integrated module may be implemented in the form of hardware, or may be implemented in the form of a software functional module. If the integrated module is implemented in the form of the software functional module and is sold or used as an independent product, the integrated module may be stored in a computer-readable storage medium. The storage medium may be a read-only memory, a magnetic disk, an optical disc, or the like.

[0114] The above descriptions are merely exemplary implementations of this application, and are not intended to limit the scope of protection of this application. Any variation or replacement readily figured out by a person skilled in the art within the technical scope recorded in this application shall fall within the scope of protection of this application. Therefore, the scope of protection of this application should be subject to the scope of

protection of the claims.

CLAIMS

1. A remote memory call method, applied to an intermediate device connected to a plurality of computing devices, the method comprising:

obtaining a memory access request from a first computing device, and searching for a memory address of borrowed memory that is allocated to the first computing device and that is from a second computing device;

accessing the memory address of the borrowed memory, and forwarding the memory access request to the second computing device through a target transmission link corresponding to the borrowed memory; and

obtaining a processing result of the borrowed memory of the second computing device for the memory access request, and forwarding the processing result to the first computing device.

2. The method according to claim 1, wherein before obtaining the memory access request from the first computing device, the method further comprises:

obtaining a memory borrowing request sent by the first computing device, wherein the memory borrowing request indicates a size of borrowed memory and a memory type of the borrowed memory;

determining the second computing device matching the memory borrowing request from at least one other computing device, and forwarding the memory borrowing request to the second computing device; and

obtaining the memory address of the borrowed memory allocated by the second computing device.

3. The method according to claim 1 or 2, wherein the obtaining the memory access request from the first computing device comprises:

obtaining the memory access request that the first computing device initiates by accessing a PCI bus register, wherein the memory access request carries a physical address of the PCI bus register; and

the searching for the memory address of the borrowed memory that is allocated to the first computing device and that is from the second computing device comprises:

searching an address mapping relationship corresponding to borrowed memory for the memory address of the borrowed memory that corresponds to the physical address of the PCI bus register carried in the memory access request, wherein the address mapping relationship is created by the intermediate device after allocating the borrowed memory.

4. The method according to claim 3, wherein before obtaining the memory access request

from the first computing device, the method further comprises:

establishing an address mapping relationship between the memory address of the borrowed memory and the physical address of the PCI bus register after obtaining the memory address of the borrowed memory that is allocated by the second computing device for a memory borrowing request of the first computing device; and

notifying the first computing device of the physical address of the PCI bus register.

5. The method according to any one of claims 1 to 4, wherein the accessing the memory address of the borrowed memory, and forwarding the memory access request to the second computing device through the target transmission link corresponding to the borrowed memory comprises:

determining the target transmission link used in a current memory borrowing process, wherein a type of the target transmission link is determined based on a memory type of the borrowed memory; and

accessing the memory address of the borrowed memory, and forwarding the memory access request to the second computing device through the determined target transmission link.

6. The method according to any one of claims 1 to 5, wherein the obtaining the processing result of the borrowed memory of the second computing device for the memory access request, and forwarding the processing result to the first computing device comprises:

obtaining the processing result that is returned by the second computing device for the memory access request through the target transmission link, and forwarding the processing result to the first computing device through a PCI bus register.

7. The method according to any one of claims 1 to 6, wherein the method further comprises:

obtaining a memory deactivation request from the first computing device; and

searching for the second computing device that provides the borrowed memory to the first computing device, and notifying the second computing device to stop lending the borrowed memory.

8. A remote memory call method, applied to a first computing device, the method comprising:

sending a memory access request to an intermediate device connected to a plurality of computing devices; and

receiving a processing result for the memory access request, the processing result being forwarded by the intermediate device, wherein the processing result is obtained by borrowed memory of a second computing device processing the memory access request, the memory access request is forwarded to the second computing device by the intermediate device

accessing a memory address of the borrowed memory and using a target transmission link corresponding to the borrowed memory for forwarding, and the memory address of the borrowed memory that is allocated to the first computing device and that is from the second computing device is searched for by the intermediate device after obtaining the memory access request of the first computing device.

9. The method according to claim 8, wherein the sending the memory access request to the intermediate device connected to the plurality of computing devices comprises:

sending the memory access request to the intermediate device by accessing a PCI bus register, wherein the memory access request carries a physical address of the PCI bus register, for the intermediate device to search an address mapping relationship corresponding to borrowed memory for the memory address of the borrowed memory that corresponds to the physical address of the PCI bus register carried in the memory access request.

10. The method according to claim 9, wherein the sending the memory access request to the intermediate device by accessing the PCI bus register comprises:

initiating the memory access request based on a virtual address of the PCI bus register in a virtual machine of the first computing device; and

translating the virtual address of the PCI bus register into a physical address in the first computing device, accessing the PCI bus register via the physical address in the first computing device, and sending the memory access request to the intermediate device.

11. The method according to claim 10, wherein before sending the memory access request to the intermediate device connected to the plurality of computing devices, the method further comprises:

translating the physical address of the PCI bus register into the virtual address after the borrowed memory is allocated by the intermediate device, and binding the virtual address to the virtual machine initiating a memory borrowing request, for the virtual machine to initiate the memory access request by accessing the PCI bus register via the virtual address.

12. The method according to claim 11, wherein the translating the physical address of the PCI bus register into the virtual address after the borrowed memory is allocated the intermediate device comprises:

obtaining the physical address of the PCI bus register that is sent by the intermediate device after the borrowed memory is allocated, and

translating the physical address into the virtual address.

13. A network interface card, connected to a plurality of computing devices, the network interface card comprising:

a memory access request obtaining module, configured to: obtain a memory access request from a first computing device, and search for a memory address of borrowed memory that is allocated to the first computing device and that is from a second computing device;

an address access module, configured to: access the memory address of the borrowed memory, and forward the memory access request to the second computing device through a target transmission link corresponding to the borrowed memory; and

a result obtaining module, configured to: obtain a processing result of the borrowed memory of the second computing device for the memory access request, and forward the processing result to the first computing device.

14. An electronic device, comprising a memory, a processor, and a computer program stored in the memory, wherein the processor implements the method according to any one of claims 1 to 12 when executing the computer program.

15. A computer-readable storage medium, having a computer program stored therein, wherein when the computer program is executed by a processor, the method according to any one of claims 1 to 12 is implemented.