1-a imposes no specific limitation on this embodiment of the present invention.

[0058] The first physical address space 160 in FIG. 1-a is a physical address space corresponding to the storage area, used to store the file system, in a physical memory. Further, as shown in FIG. 1-b, the first physical address space 160 used to store the file system may further be divided into: a first physical memory area 161 used to store metadata and data that are of the file system, and a second physical memory area 162 used to store buffered data generated during running of the file system. The buffered data generated during running of the file system specifically refers to data generated during file access by a process, such as a file object "file", a file index node object "inode", a file directory object "dentry", and a file descriptor (which may also be referred to as a handle). It should be understood that, the second physical address space 170 that is used to store the target file and that is in the first physical address space 160 is in the first physical memory area 161 (shown in FIG. 1-b). Optionally, the first physical memory area 161 and the second physical memory area 162 may be not contiguous in terms of physical address.

[0059] As shown in FIG. 1-b, the first mapping relationship exists between the first virtual address space 140 and the first physical address space 160. Therefore, corresponding to the two areas obtained through division of the first physical address space 160, the first virtual address space 140 may also correspondingly include two areas: a first area 141 and a second area 142. A mapped area, in the first virtual address space 140, of the first physical memory area 161 is the first area 141, and a mapped area, in the first virtual address space 140, of the second physical memory area 162 is the second area 142. It should be understood that, a process can access the first physical memory area 161 and the second physical memory area 162 respectively by accessing the first area 141 and the second area 142, so as to access the metadata and the data that are of the file system, and the buffered data generated during running of the file system. It should be understood that, the mapped address space, in the first virtual address space 140, of the second physical address space 170 that stores the target file is in the first area 141.

[0060] It may be understood that, the division of the first physical address space 160 into two areas is only an example. Alternatively, in an actual application, the first physical address space 160 may be divided into multiple areas according to an actual need, and certainly, in an actual application, the first virtual address space 140 may also be divided into multiple areas according to an actual need, which are not limited herein.

[0061] In the embodiments of the present invention, in one case, the first physical memory area 161 and the second physical memory area 162 both may use a non-volatile memory (Non-Volatile Memory, NVM) medium, such as a phase change memory (Phase Change Memory, PCM), a variable resistive random access memory (Resistive RAM, ReRAM), a magnetic random

access memory (Magnetic Random Access Memory, MRAM), or another non-volatile storage medium that has a read/write feature similar to that of a DRAM and that can be mounted on a memory bus for access. In another case, the first physical memory area 161 may use a non-volatile storage medium; the second physical memory area 162 may use a volatile storage medium, specifically, such as a dynamic random access memory (Dynamic Random Access Memory, DRAM) medium with abrasion resistance.

[0062] A file access method 200 in an embodiment of the present invention may be executed by a controller or a central processing unit (Central Processing Unit, CPU).

[0063] The following details file access methods according to the embodiments of the present invention with reference to FIG. 2 to FIG. 5. FIG. 2 is a schematic flowchart of a file access method according to an embodiment of the present invention. It should be noted that, in FIG. 2, a file access request of a process is used as an example for description. It may be understood that, the method 200 may be applied to any scenario in which a process requests to access a file. As shown in FIG. 2, the method 200 includes:

[0064] S210: Receive a file access request from a process, where the file access request includes a file identifier, and the file identifier is used to indicate a to-be-accessed target file.

[0065] Specifically, the file identifier may include a storage path of the target file, a file type of the target file (a directory file or a data file), and the like. It should be understood that, the file access request may be specifically a file open request, a file read request, a file write request, or another access request.

[0066] S220: Access a first physical address space according to a preset first virtual address space and a preset first mapping relationship between the first virtual address space and the first physical address space, where the first virtual address space is a part of user space in an operating system, and the first physical address space stores a file system and is a part of memory space.

[0067] Optionally, in this embodiment of the present invention, S220 of accessing a first physical address space according to a preset first virtual address space and a preset first mapping relationship between the first virtual address space and the first physical address space, where the first physical address space stores a file system, includes:

[0068] S221: Access the first virtual address space according to the file access request.

[0069] S222: Determine that a process page table corresponding to the first virtual address space records the first mapping relationship between the first virtual address space and the first physical address space that stores the file system.

[0070] S223: Access the first physical address space according to the first virtual address space and the first mapping relationship.

information, included in the index node, of the target file, where the file page table records a second physical address space in the first physical address space, a data page of the target file is stored in the second physical address space, and the data page of the target file is used to store file data.

[0080] Specifically, the file page table of the target file is used to point to a physical address, in the first physical address space, of the memory page that stores the target file. Therefore, the second physical address space of the target file can be oriented according to the file page table of the target file.

[0081] Optionally, in this embodiment of the present invention, a data structure of the file page table of the target file is similar to or the same as that of the process page table.

[0082] It should be understood that, in the prior art, the process page table is used by a process in the operating system in protection mode, and represents a mapping relationship between virtual space and physical space that are of a process. For a given to-be-accessed virtual address, an MMU orients, in the current process page table according to the virtual address, a process page table entry corresponding to the virtual address, and then parses the page table entry to obtain an actual physical address, so as to implement access. There is different addressing space of the operating system, including 32-bit addressing space with a size of 2^{32} bytes, or 64-bit addressing space with a size of 2^{64} bytes. A common process page table may be a three-level page table or a four-level page table. The four-level page table includes a PGD, a PUD, a PMD, and a PTE. The three-level page table is compatible with the four-level page table, and also includes a PGD, a PUD, a PMD, and a PTE nominally. However, the PUD and the PMD belong to a same level of a page table. A higher-level page table entry includes a physical start address pointing to a lower-level page table entry, and the last-level page table entry PTE includes a physical start address pointing to a specific physical page.

[0083] In this embodiment of the present invention, the structure of the file page table may be the same as or similar to that of the process page table. Specifically, the file page table may be a single-level or multi-level page table, where each entry of each level of page table in a non-lowest page table records a physical start address of a lower-level page table, and each entry of a lowest-level page table records a physical start address of a memory page of a file. Specifically, as shown in FIG. 3, FIG. 3 gives an example in which the file page table of the target file is a three-level file page table including a PUD, a PMD, and a PTE. Each entry in a PUD records a physical start address of a PMD, each entry of a PMD records a physical start address of a PTE, and each entry of a PTE records a physical start address of a memory page (a memory page stores file content of the target file). It should be understood that, not all files need to use a three-level file page table, and even if a three-level page table is used, the page table does not definitely have a fixed format. For example, not all files need to use 1024 PTEs, and an organization structure of a

in the second physical address space with reference to FIG. 4.

[0090] As shown in FIG. 4, optionally, in the file access method 200 shown in FIG. 2, S250 of accessing the target file according to the second physical address space includes:

[0091] S251: Allocate the second virtual address space to the target file, where the second virtual address space is a part of private space, allocated to the process, in the user space in the operating system, and the second virtual address space is a contiguous address space.

[0092] Specifically, the second virtual address space may be a segment of contiguous virtual address space in the file access space 151 of the private process space 150 described above with reference to FIG. 1-a.

[0093] S252: Establish the second mapping relationship between the second physical address space and the second virtual address space.

[0094] Optionally, in this embodiment of the present invention, S252 of establishing the second mapping relationship between the second physical address space and the second virtual address space includes:

establishing the second mapping relationship by linking the file page table of the target file to a process page table corresponding to the second virtual address space.

[0095] Specifically, it is assumed that the process page table corresponding to the second virtual address space is an N-level page table while the file page table of the target file is an M-level page table, where $M \leqq N$, and the file page table lacks page table entries of level 0 to level (N-M-1). The linking the file page table of the target file to a process page table corresponding to the second virtual address space specifically refers to: copying highest (N-M) levels of page table entries of the file page table to (N-M) levels of page table entries of the process page table corresponding to the second virtual address space. According to this manner, the second mapping relationship can be established between the second virtual address space and the second physical address space of the target file. It can be learnt from the foregoing descriptions that, to establish the second mapping relationship in a "linking" manner, a highest-level page table entry of the file page table of the target file needs to be copied to a corresponding level of page table entry of the process page table corresponding to the second virtual address space. In this way, other page entries whose levels are lower than that of the highest-level (level (N-M)) page table entry of the file page table of the target file are automatically linked to the process page table corresponding to the second virtual address space. Therefore, in this "linking" manner, a quantity of copied page tables can be effectively reduced, and software running overheads can be reduced.

[0096] After the second mapping relationship is established between the second physical address space of the target file and the second virtual address space, the second physical address

second virtual address space, an access address definitely is included in "the virtual address space corresponding to the target file", and a mapping relationship exists between each virtual address in "the virtual address space corresponding to the target file" and a specific physical address (a physical address of the data page of the target file); therefore, in the foregoing case, the "data page fault" does not occur. (2) If the second virtual address space is greater than "the virtual address space corresponding to the target file", when the target file is accessed by using the second virtual address space, the foregoing "data page fault" occurs because an access address is not included in "the virtual address space corresponding to the target file".

[0103] In the case in which the second virtual address space is greater than "the virtual address space corresponding to the target file", when a read and/or write operation is performed on the target file by using the second virtual address space, out-of-bound access may occur. To reduce a read error caused by an out-of-bound read operation, in this embodiment of the present invention, S253 of accessing the target file in the second physical address space according to the second virtual address space and the second mapping relationship may include:

performing a read operation on the target file in the second physical address space according to the second virtual address space and the second mapping relationship; and

when a virtual address corresponding to data to be read through the read operation is not included in the virtual address space corresponding to the target file, ending the read operation.

[0104] It should be understood that, "the virtual address space corresponding to the target file" in this embodiment refers to the foregoing mapped space that is obtained by mapping the second physical address space of the target file to the second virtual address space and that is a part of the second virtual address space.

[0105] Specifically, a file read instruction used for requesting to read the file data of the target file is received from the process; based on the second mapping relationship, the second physical address space is accessed by using the second virtual address space, to perform the read operation on the target file; when a virtual address corresponding to data to be read through the read operation is not included in the virtual address space corresponding to the target file, the read operation ends. Specifically, when an end read address that is obtained after a start read address in the second virtual address space offsets a preset read byte length is not included in "the virtual address space corresponding to the target file", the read operation performed on the target file ends.

[0106] Optionally, in this embodiment of the present invention, a file descriptor "fd" of the target file and a file object “file” of the target file may be created in the second physical memory area 162 (shown in FIG. 1-b) of the first physical address space 160, and an association relationship between the file descriptor "fd" and the file object “file” of the target file is created. The file object

"file" is used to record a size of the second virtual address space, a start address that is obtained by mapping the second physical address space of the target file to the second virtual address space, a file size of the target file, and other information. When the read operation request for the target file is received, the file descriptor "fd" of the target file in the second physical memory area 162 may be accessed by using a virtual address (a mapped address, in the second area 142, of a physical address that stores the file descriptor "fd" and that is in the second physical memory area 162), corresponding to the file descriptor "fd" of the target file, in the second area 142 (shown in FIG. 1-b) of the first virtual address space, so as to obtain the size of the second virtual address space allocated to the target file, the start address that is obtained by mapping the second physical address space of the target file to the second virtual address space, the file size of the target file, and the other information, and further to perform a read operation on the target file by using the second virtual address space.

[0107] Therefore, in the file access method according to this embodiment of the present invention, a problem of an out-of-bound read operation can be resolved, which can effectively ensure access security.

[0108] To reduce a write error caused by an out-of-bound write operation, in this embodiment of the present invention, if the second virtual address space is greater than the second physical address space, S253 of accessing the target file in the second physical address space according to the second virtual address space and the second mapping relationship includes:

performing a write operation on the target file in the second physical address space according to the second virtual address space and the second mapping relationship; and

when a virtual address corresponding to data to be written through the write operation is not included in the second virtual address space, ending the write operation.

[0109] Specifically, a file write request for requesting to perform the write operation on the target file is received from the process; based on the second mapping relationship, the second physical address space is accessed by using the second virtual address space, to perform the write operation on the target file; when a virtual address corresponding to data to be written through the write operation is not included in the second virtual address space, the write operation ends. Specifically, when an end write address that is obtained after a start write address in the second virtual address space offsets a preset write byte length is not included in the second virtual address space, the write operation performed on the target file ends.

[0110] Optionally, in this embodiment of the present invention, a file descriptor "fd" of the target file and a file object "file" of the target file may be created in the second physical memory area 162 (shown in FIG. 1-b) of the first physical address space 160, and an association relationship

between the file descriptor "fd" and the file object “file” of the target file is created. The file object “file” is used to record a size of the second virtual address space, a start address that is obtained by mapping the second physical address space of the target file to the second virtual address space, a file size of the target file, and other information. When the write operation request for the target file is received, the file descriptor "fd" of the target file in the second physical memory area 162 may be accessed by using a virtual address (a mapped address, in the second area 142, of a physical address that stores the file descriptor "fd" and that is in the second physical memory area 162), corresponding to the file descriptor "fd" of the target file, in the second area 142 (shown in FIG. 1-b) of the first virtual address space, so as to obtain the size of the second virtual address space allocated to the target file, the start address that is obtained by mapping the second physical address space of the target file to the second virtual address space, the file size of the target file, and the other information, and further to perform a write operation on the target file by using the second virtual address space. As described above, in the case in which the second virtual address space is greater than "the virtual address space corresponding to the target file", the "data page fault" may occur when the write operation is performed on the target file by using the second virtual address space. After interruption processing is complete, the write operation may continue to be performed on the target file.

[0111] Optionally, in this embodiment of the present invention, during a process of performing the write operation on the target file in the second physical address space according to the second virtual address space and the second mapping relationship:

when the process page table corresponding to the second virtual address space lacks an entry recording a mapped physical address, in the first physical address space, of a current write address, the write operation performed on the target file is interrupted, where the current write address is not included in the virtual address space corresponding to the target file;

a third physical address is allocated to the target file in the first physical address space, and an entry that is used to record a mapping relationship between the current write address and the third physical address is added to the process page table corresponding to the second virtual address space; and

based on the mapping relationship recorded in the foregoing entry that is added to the process page table corresponding to the second virtual address space, the third physical address is accessed by using the current write address, and the write operation continues to be performed on the target file.

[0112] In the file access method according to this embodiment of the present invention, a range of a file write operation can be increased, and a problem of out-of-bound access can be resolved,

[0125] S370: Obtain a file page table of the target file according to the file page table information, included in the index node, of the target file, where the file page table records a second physical address space in the first physical address space, a data page of the target file is stored in the second physical address space, and the data page of the target file is used to store file data. S370 is the same as S240 shown in FIG. 2, and details are not described herein again.

[0126] S380: Access the target file according to the second physical address space. S380 is the same as S250 shown in FIG. 2, and details are not described herein again.

[0127] In the file access method according to this embodiment of the present invention, if the process page table corresponding to the first virtual address space does not record the first mapping relationship, a page fault needs to be executed one time. That is, the user space is switched to the kernel space, and the mapping page table corresponding to the first mapping relationship is linked to the process page table corresponding to the first virtual address space. If the process page table corresponding to the first virtual address space records the first mapping relationship, the first physical address space that stores the file system is accessed according to the first mapping relationship and the first virtual address space, so that a file in the file system is accessed. It should be noted that, in this embodiment of the present invention, the first virtual address space is space, shared by all processes, in the user space. For file access requests of all processes, access to a requested target file can be implemented directly in the user space.

[0128] It should be understood that, in this embodiment of the present invention, after the mapping page table corresponding to the first mapping relationship between the first virtual address space and the first physical address space is linked to the process table corresponding to the first virtual address space, if the system is not shut down, the process page table corresponding to the first virtual address space permanently stores the entry recording the first mapping relationship, and any process can subsequently implement, based on the first virtual address space, file access in the user space.

[0129] Compared with the prior art, in the file access method according to this embodiment of the present invention, file access can be implemented in user space when a quantity of system calls is reduced to a relatively great extent. Therefore, software running overheads can be reduced effectively during file access, and a file access speed can be increased effectively and file access efficiency can be improved effectively.

[0130] It should be understood that, the technical solution provided in this embodiment of the present invention may be applied to a Linux operating system, or may be applied to another operating system in which each process has its dedicated space.

[0131] The foregoing details the file access methods according to the embodiments of the present invention with reference to FIG. 1 to FIG. 5, and the following will detail a file access

[0145] Optionally, the processor 510 invokes, by using the bus system 530, the code stored in the memory 520, to specifically implement: establishing the second mapping relationship by linking the file page table of the target file to a process page table corresponding to the second virtual address space.

[0146] Optionally, the second virtual address space is greater than the second physical address space. The processor 510 invokes, by using the bus system 530, the code stored in the memory 520, to specifically implement: performing a read operation on the target file in the second physical address space according to the second virtual address space and the second mapping relationship; and when a virtual address corresponding to data to be read through the read operation is not included in a virtual address space corresponding to the target file, ending the read operation.

[0147] Optionally, the second virtual address space is greater than the second physical address space. The processor 510 invokes, by using the bus system 530, the code stored in the memory 520, to specifically implement: performing a write operation on the target file in the second physical address space according to the second virtual address space and the second mapping relationship; and when a virtual address corresponding to data to be written through the write operation is not included in the second virtual address space, ending the write operation.

[0148] Optionally, the processor 510 invokes, by using the bus system 530, the code stored in the memory 520, to further implement: establishing, in kernel space, the first mapping relationship between the first virtual address space and the first physical address space, and storing, in a process page table corresponding to the kernel space, a mapping page table corresponding to the first mapping relationship; after the receiving the file access request from the process, determining that a process page table corresponding to the first virtual address space does not record the first mapping relationship; and linking, to the process page table corresponding to the first virtual address space, the mapping page table that is corresponding to the first mapping relationship and that is stored in the kernel space.

[0149] It should be understood that, in this embodiment of the present invention, the processor 510 may be a central processing unit (Central Processing Unit, "CPU" for short), or the processor 510 may be another general purpose processor, a digital signal processor (DSP), an application-specific integrated circuit (ASIC), a field programmable gate array (FPGA) or another programmable logic device, a transistor logic device, a hardware component, or the like. The general purpose processor may be a microprocessor, or the processor may be any regular processor or the like.

[0150] The memory 520 may include a read-only memory and a random access memory, and provides an instruction and data for the processor 510. The memory 520 may further include a