一种计算机设备、异常处理的方法以及中断处理的方法

本申请要求于 2020 年 8 月 19 日提交中国专利局、申请号为 202010839905.3、发明名称为“一种计算机设备、异常处理的方法以及中断处理的方法”的中国专利申请的优先权，其全部内容通过引用结合在本申请中。

技术领域

本申请涉及计算机技术领域，具体涉及一种计算机设备、异常处理的方法以及中断处理的方法。

背景技术

在虚拟化系统中，实现输入/输出（input/output，I/O）虚拟化主要采用虚拟设备模拟方案。虚拟设备模拟方案根据对虚拟 I/O 的处理方式可分为同步处理和异步处理两种。

在同步处理过程中，虚拟机（virtual machine，VM）执行与虚拟 I/O 设备相关的操作，（如：对串口内存的读操作或写操作，或者对其他虚拟 I/O 设备的访问操作）会触发异常指令。VM 向位于宿主机（host）的内核态的虚拟机监视器（virtual machine monitor，VMM）发送异常指令，由 VMM 将该异常指令传递到 host 的用户态，进而在 host 的用户态进行异常处理。处理完异常后，将处理结果从 host 的用户态传递到 VMM，再从 VMM 传递到 VM。该过程会出现了两次 VM 与 VMM 的切换开销，两次 host 内核与用户态的切换开销。

在异步处理过程中，VM 执行对磁盘的内存或网卡的内存的读操作或写操作，会触发中断请求。该中断请求需要通过异步的方式通知远端处理器上的快速仿真器（quick emulator，Qemu）线程完成对该中断请求的处理。该远端处理器为不同于支撑该 VM 运行的本地处理器之外的另一处理器。在本地处理器，VM 会触发异常指令，然后 VM 将该异常指令发送给 host 的内核态的 VMM，然后 VMM 识别是用户态的异常指令，VMM 会将该异常指令从 host 的内核态传递到 host 的用户态，接下来再从 host 的用户态触发系统调用进入 host 的内核态，然后在 host 的内核态触发中断请求，并向远端处理器传递该中断请求，远端处理器的 Qemu 线程需要将处于内核态的调度器唤醒，这个过程需要内核态到用户态的切换，然后再通过系统调用进入内核态才能实现磁盘等物理设备的访问。可见，异步处理的过程也需要频繁进行用户态与内核态的切换。

无论是同步处理过程，还是异步处理过程，都会产生多次切换开销，影响了 I/O 的性能。

发明内容

本申请实施例提供一种计算机设备、异常处理的方法以及中断处理的方法，用于减少切换开销，提高计算机设备的性能。本申请还提供了相应的装置、计算机可读存储介质、计算机程序产品等。

本申请第一方面提供一种计算机设备，该计算机设备包括硬件层，硬件层上运行有宿主机host，硬件层包括第一处理器、第一处理器上运行有虚拟机（virtual machine，VM），第一处理器包括重定向装置；第一处理器用于：获取VM触发的异常指令；重定向装置被配置为：从VM的虚拟化特权级切换到host的用户态；第一处理器还用于：在host的用户态下执行异常指令对应的异常处理函数。

该第一方面通过同步处理的方式实现了对VM触发的异常的处理。

需要说明的是，本申请实施例中的处理器可以是处理核，一个处理核一个时间段内支持一个VM运行。多个VM可以通过分时复用的方式使用同一个处理核。

该第一方面中，计算机设备可以为手机、平板电脑、桌面型计算机、笔记本电脑、超级移动个人计算机（ultra-mobile personal computer，UMPC）、手持计算机、上网本等终端设备，该计算机设备也可以是服务器。硬件层可以包括处理器、存储器和通信接口等硬件设备。VM和host都是通过软件模拟出来的设备，VM和host都运行在硬件层上。第一处理器包括的重定向装置可以是通过硬件电路来实现的。该异常指令可以是VM执行输入/输出（input/output,I /O）操作时触发的，该I/O操作可以是VM要访问虚拟I/O设备触发的操作，通常，虚拟I/O设备包括前端驱动和后端设备，前端驱动运行在VM中，后端设备的程序运行在host中，若VM执行的操作要通过前端驱动访问后端设备，则会触发异常指令，通过该异常指令实现对后端设备的访问。虚拟I/O设备是后端设备的一种，访问虚拟I/O设备可以是要从虚拟I/O设备的内存中读取数据或者向该虚拟I/O设备的内存写入数据。虚拟I/O设备是虚拟机能使用的I/O设备，该虚拟I/O设备可以是通过软件模拟出来的纯虚拟化的I/O设备，该虚拟I/O设备也可以通过映射的方式映射到物理I/O设备上，访问该虚拟I/O设备，最终会访问到物理I/O设备。

该异常指令可以是第一VM在执行超出第一VM权限的操作时发生了异常，第一处理器可以通过调用存储器中存储的计算机指令，执行相应的检测逻辑检测到该异常代码，该步骤也可以是第一处理器执行硬件检测逻辑检测到该异常代码，进而获取到该异常指令。第一处理器在获取异常指令后，可以将该异常产生的原因、异常指令的类型，第一VM所要访问的内存，以及返回地址都写入到硬件寄存器中，供后续处理异常使用。

VM的虚拟化特权级表示处于虚拟化模式下的特权级，也就是VM的工作模式，也可称为虚拟化模式。该虚拟化特权级包括虚拟化的用户态和虚拟化的内核态。从VM的虚拟化特权级切换（直接切换）到host的用户态可以包括将VM的虚拟化特权级的信息从用于存储特权级的寄存器中移出，然后将host的用户态的信息写入到该寄存器的过程，该寄存器可以是系统寄存器。本申请中特权级和虚拟化可以分别用1个比特位来表示，例如：用0表示用户态，用1来表示内核态，用0来表示非虚拟化，用1来表示虚拟化。特权级在第一个比特位，是否虚拟化在第二个比特位，例如：00表示非虚拟化的用户态，也可以称为宿主机的用户态，10示非虚拟化的内核态，也可以称为宿主机的内核态，01表示虚拟化的用户态，也可以称为VM的用户态，11表示虚拟化的内核态，也可以称为VM的内核态。

用户态的host指的是工作在用户态的宿主机，host的用户态指的是宿主机的一个工作模式，宿主机的工作模式包括用户态和内核态。

重定向装置被配置为从VM的虚拟化特权级切换到host的用户态，可以理解为重定向装置用于切换第一处理器的工作模式，将第一处理器的工作模式从虚拟化模式切换到非虚拟化模式下的用户态（host的用户态）。处理器的工作模式可以理解为处理器的工作状态，处理器的工作状态可以通过上述系统寄存器中的比特位的值来表示。

异常处理函数可以包含于运行在host的用户态的虚拟处理器(virtual central

processing unit ， vCPU ）的线程中。第一处理器可以在该host的用户态下执行异常指令对应的异常处理函数。异常指令与异常处理函数之间可以通过异常类型来关联，可以是不同的异常类型对应不同的异常处理函数，如：VM在执行对串口的写操作时对应的异常处理函数与VM在执行对显卡的写操作时对应的异常处理函数不同，也可以是所有的异常指令对应同一个异常处理函数，该在执行该异常处理函数时会通过调用不同的数据来执行对不同异常指令的处理。例如：若该异常指令是VM执行I/O操作触发的，则执行该异常处理函数就会完成相应的I/O操作，如：读取I/O设备的内存中的数据或向该I/O设备的内存写入数据。若该异常指令是VM执行其他操作（如：缺页异常）触发的，则第一处理器会执行该异常处理函数来完成缺页异常的处理过程。换句话说，重定向装置被配置为从虚拟化模式切换至非虚拟化模式。因此用户态的中断或异常可以被重定向至host的用户空间进行处理，即由用户态的host执行异常或中断处理函数。

由该第一方面可知，该计算机设备可以通过重定向装置将异常指令从VM的虚拟化特权级直接切换到host的用户态，在该host的用户态执行异常处理函数。相比于现有技术需要先从VM的虚拟化特权级切换到host的内核态，再从host的内核态切换到host的用户态相比，可以降低切换开销，提高计算机设备的性能。

在第一方面的一种可能的实现方式中，重定向装置还被配置为：将异常处理函数的入口地址写入到第一寄存器中；第一处理器，具体用于从第一寄存器中读取异常处理函数的入口地址，并根据入口地址调用异常处理函数，在host的用户态下执行异常指令对应的异常处理函数。

该种可能的实现方式中，重定向装置可以将异常处理函数（可称做trap handler）的入口地址可写入到第一寄存器中，这样第一处理器接下来就会从该第一寄存器中读取到该入口地址，进而通过该入口地址调用该异常处理函数，在host的用户态下执行该异常处理函数。重定向装置中的“重定向”可以包括特权级的切换，以及将异常处理函数的入口地址写入到第一寄存器中的过程。该种可能的实现方式中，通过重定向装置完成入口地址在第一寄存器中的写入，可以使第一处理器快速执行异常处理函数。

在第一方面的一种可能的实现方式中，该异常处理函数包含于运行在host的用户态的虚拟处理器的线程中，虚拟处理器的线程被分配的执行环境包括主线程执行环境、VM的执行环境和影子执行环境，虚拟处理器的线程被分配的执行环境为虚拟处理器的线程运行所需的资源，影子执行环境具有独立于主线程执行环境和VM的执行环境的用户态内存空间；影子执行环境用于执行异常处理函数，异常处理函数用于处理触发异常指令的异常；第一处理器用于：从VM的上下文切换为用户态的影子执行环境的上下文，影子执行环境的用户态内存空间用于存储在切换上下文时移出的VM的上下文。

本申请中执行异常处理函数包括切换上下文和在host的用户态下处理异常的过程。

该种可能的实现方式中，虚拟处理器的线程在运行前会向宿主机申请资源，宿主机为该虚拟处理器的线程分配运行所需要的资源。该资源可以称为执行环境。本申请中的“执行环境”包括软件执行环境和硬件执行环境两个方面，软件执行环境包括虚拟处理器的线程运行在不同执行环境时所使用的栈中的信息，以及线程的上下文的信息，硬件方面包括

线程运行时所使用的硬件资源，包括内存资源或处理资源等。主线程执行环境、VM的执行环境和影子执行环境都有各自独立的内存空间，其中，主线程执行环境包括用户态的主线程的执行环境，内核态的主线程的执行环境。内核态的主线程的执行环境用于存储VM和host的上下文。影子执行环境包括用户态的影子执行环境和内核态的影子执行环境。用户态的影子执行环境也就是影子执行环境的用户态内存空间。该用户态内存空间表示影子执行环境在用户态所对应的内存地址空间，该用户态内存空间用于存储在切换上下文时从寄存器移出的VM的上下文。因为影子执行环境具有独立的用户态内存空间，该用户态内存空间可以保存VM的上下文，不需要借助主线程执行环境先从VM的虚拟化特权级切换到host的内核态，再从内核态切换到用户态，这样可以进一步减少切换开销，提高计算机设备的性能。

在第一方面的一种可能的实现方式中，所述第一处理器还用于，在执行异常处理函数时产生需要在host的内核态处理的中断或异常，则从所述用户态的影子执行环境的上下文切换到内核态的影子执行环境的上下文；在host的内核态处理需要在host的内核态处理的中断或异常；从所述内核态的影子执行环境的上下文切换回所述用户态的影子执行环境的上下文继续执行异常处理函数。

该种可能的实现方式中，影子执行环境还包括内核态的影子执行环境，也就是内核态用户空间，在从用户态的影子执行环境的上下文切换到内核态的影子执行环境的上下文时，该内核态用户空间用于保存从寄存器切换出的用户态的影子执行环境的上下文。该种可能的实现方式保证了在执行异常处理函数时发生需要在内核态执行的中断或异常时的处理，提高了计算机系统的兼容性。

本申请第二方面提供一种计算机系统，该计算机系统在上述第一方面所提供的计算机系统上，还包括第二处理器和中断控制器。上述第一方面中在执行异常处理函数之前的过程都可以参阅第一方面中的内容进行理解。不同的是，第一处理器不执行该异常处理函数，而是向中断控制器发送访问请求，访问请求指示访问中断控制器中的用户态地址映射空间。然后，中断控制器向第二处理器发送中断请求；第二处理器根据该中断请求执行异常处理函数。

该第二方面，借助第一方面的同步处理方案中的重定向装置实现重定向到host的用户态，然后通过访问中断控制器中的用户态地址映射空间进而实现异步处理的过程。该第二方面借助重定向装置实现了从VM的特权级到host的用户态的切换，减少了异步处理的切换开销，提高了计算机设备的性能。

本申请第三方面提供一种计算机设备，该计算机设备包括硬件层，硬件层上运行有宿主机host，硬件层包括第一处理器、第二处理器和中断控制器，第一处理器上运行有虚拟机第一VM，第一处理器包括虚拟事件通知装置，第一处理器用于：获取第一VM触发的第一中断请求，该第一中断请求用于请求执行中断处理函数；虚拟事件通知装置被配置为：响应于第一中断请求，直接向中断控制器发送中断信息；中断控制器被配置为：根据中断信息确定第二处理器的标识，根据第二处理器的标识向第二处理器发送第二中断请求；第二处理器用于：响应于接收到的第二中断请求，执行所述中断处理函数。

该第三方面通过异步处理的方式实现了对第一VM触发的中断的处理，异步处理指的是

跨处理器的处理过程，即运行在第一处理器上的第一VM执行中断，需要通过第二处理器才能完成该中断处理的过程。

该第三方面中，虚拟事件通知装置可以是通过硬件电路来实现的。该第一中断请求可以是第一VM执行I/O操作时触发的，该I/O操作可以是第一VM要访问虚拟I/O设备触发的操作。中断信息用于指示第一VM中发生的虚拟事件。该第三方面提供的采用异步方式处理中断的计算机设备，通过虚拟事件通知装置可以不需要进入host的内核态就可以实现对中断事件的处理，减少了特权级的切换开销，提高了计算机设备的性能。

在第三方面的一种可能的实现方式中，第一中断请求包括虚拟事件中断请求号VEI ID，VEI ID用于标识触发第一中断请求的虚拟事件；第一处理器还用于：获取第一VM的VM ID；中断信息包括VEI ID和第一VM的VM ID；中断控制器具体被配置为：根据VEI ID和第一VM的VM ID确定对应的第二处理器的标识、VEI物理号，以及处理第一中断请求的host的特权级，VEI物理号用于在host中标识虚拟事件；根据第二处理器的标识向第二处理器发送第二中断请求，该第二中断请求包括指示host的特权级的信息；第二处理器具体用于：获取VEI物理号，根据host的特权级对应的处理方式，处理第二中断请求，执行与所述VEI物理号对应的中断处理函数。

该种可能的实现方式中，VEI ID是用于标识虚拟机中发生的虚拟事件，VEI物理号在host中指示虚拟事件的标识。VEI ID与VEI物理号具有映射关系，相互映射的VEI ID与VEI物理号指示同一虚拟事件（如：读/写磁盘的事件，读/写网卡的事件等），VEI ID可以通过数值来表示，如：0至63，每个数值标识一个不同的虚拟事件，一个I/O设备上可以有多个不同的虚拟事件，如：对读磁盘的事件可以用数值1来表示，写磁盘的事件可以用数值2来表示，针对磁盘的某个控制事件可以用数值3来表示，读网卡的事件可以用数值4来表示，写网卡的事件可以用数值5来表示等。每个虚拟机中的同一虚拟事件可以使用相同的VEI ID 。VEI物理号可以是全局唯一的，在不同的虚拟机上发生的同一VEI ID的虚拟事件可以对应不同的VEI物理号。例如：VM1的VEI ID的取值范围对应的VEI物理号可以取0至63，VM2的VEI ID的取值范围对应的VEI物理号可以取64至127，以此类推，这样，第二处理器根据不同的VEI物理号所在的范围就可以确定发生虚拟事件的虚拟机，然后再根据VEI物理号的具体取值就可以确定该VM上所发生的虚拟事件，例如：VEI ID=5时表示读磁盘的事件，那么VEI物理号=5，就表示VM1上发生了读磁盘的事件，若VEI物理号=69，则表示VM2上发生了读磁盘的事件。关于虚拟事件与VEI ID的对应关系，以及VEI物理号与虚拟机的对应关系可以根据需求设置，本申请中对此不做限定。

本申请中，因为每个虚拟机中的同一虚拟事件可以使用相同的VEI ID，为了区分该虚拟事件所发生的虚拟机，第一处理器可以获取VM ID，VM ID可以是第一处理器从用于存储虚拟机标识的寄存器中读取的。然后虚拟事件通知装置向中断控制器发生VEI ID和VM ID，中断控制器可以通过VEI ID和VM ID查找第二处理器的标识和VEI物理号，以及对应host的特权级。中断控制器中可以维护有输入是：（VEI ID）+（VM ID）->输出是：（CPU ID）+（VEI物理号）+（host的特权级）的映射表。中断控制器可以通过该映射表直接确定出对应的（CPU ID）+（VEI物理号）+（host的特权级）。中断控制器根据CPU ID可以确定接下

来执行该异步中断处理的第二处理器，根据该host的特权级可以确定接下来是采用用户态的处理方式还是内核态的处理方式。第二处理器可以从存储该VEI物理号的寄存器中读取该VEI物理号，通过该VEI物理号可以确定与该VEI物理号对应的中断处理函数，然后按照用户态的处理流程或者内核态的处理流程执行该中断处理函数。在计算机设备中可以通过一块内存来维护VEI物理号与中断处理函数的映射关系，第二处理器可以根据该VEI物理号从该内存中读取对应的中断处理函数。该种可能的实现方式，通过VEI ID和VM ID可以快速确定发生在虚拟机上的虚拟事件，进而通过虚拟事件通知装置和中断控制器可以不需要进入host的内核态就可以实现对中断事件的处理，减少了特权级的切换开销，提高了计算机设备的性能。

在第三方面的一种可能的实现方式中，第一中断请求包括虚拟事件中断请求号VEI ID，VEI ID用于标识第一VM触发第一中断请求的虚拟事件；中断信息包括VEI ID；中断控制器具体被配置为：根据VEI ID确定对应的第二处理器的标识，以及处理第一中断请求的host的特权级；根据第二处理器的标识向第二处理器发送第二中断请求，所述第二中断请求包括指示所述host的特权级的信息；第二处理器具体用于：获取所述VEI ID，根据host的特权级对应的处理方式，处理第二中断请求，以执行与VEI ID对应的操作。

该种可能的实现方式中，VEI ID可以是全局唯一的，不同的VM针对同一虚拟事件使用不同的VEI ID，这样就不需要再使用VM ID，也不需要再转换VEI 物理号，只要根据该VEI ID确定对应的第二处理器的标识以及host的特权级即可执行后续的中断处理。第二处理器可以从存储VEI ID的寄存器中获取该VEI ID。该种可能的实现方式通过VEI ID就可以快速确定发生在虚拟机上的虚拟事件，进而通过虚拟事件通知装置和中断控制器可以不需要进入host的内核态就可以实现对中断事件的处理，减少了特权级的切换开销，提高了计算机设备的性能。

在第三方面的一种可能的实现方式中，host的特权级为用户态；中断控制器还可以设置影子线程的信息。第二处理器还包括硬件辅助切换装置；硬件辅助切换装置被配置为：从第一线程切换至影子线程，该第一线程为第二处理器切换前运行的线程，影子线程是运行在用户态的host中的线程，影子线程用于执行所述中断处理函数；第二处理器具体用于，在host的用户态下执行中断处理函数。

该种可能的实现方式中，第一线程也可以称为被中断的线程，也可以是将被中断的线程的信息切换为影子线程的信息，该被中断的线程为第二处理器切换前运行的线程；第二处理器，还用于跳转到影子线程的中断处理函数的入口地址，在host的用户态执行中断处理函数，该中断处理函数与第二中断请求对应。

该种可能的实现方式中，硬件辅助切换装置可以是通过硬件电路来实现的。切换线程可以包括切换线程的信息，该第一线程的信息包括该第一线程的内存地址空间，页表首地址、数据结构信息，处理特权级等。影子线程是运行在用户态的host中的线程，用于执行中断处理函数。影子线程的信息包括影子线程的内存地址空间，页表首地址、数据结构信息，处理特权级等。影子线程可以理解为是第一线程的影子，第二处理器正在运行的第一线程被中断后，第二处理器运行影子线程来执行中断处理。该种可能的实现方式中，不需

要陷出到host的内核态，通过硬件辅助切换装置可以快速切换第一线程的信息和影子线程的信息，从而可以直接进入host的用户态处理该中断，进一步减少了切换开销，提高了计算机设备的性能。

在第三方面的一种可能的实现方式中，第二处理器还包括重定向装置，第二处理器，还用于在host的用户态下执行中断处理函数之前，确定第一线程是否处于第二VM的虚拟化特权级，第二VM运行在第二处理器上；重定向装置被配置为：若被中断的线程处于第二VM的虚拟化特权级，则将第二VM的虚拟化特权级切换为host的用户态。

该种可能的实现方式中，第二处理器中的重定向装置可以是通过硬件电路实现的，该重定向装置的功能可以参阅上述第一方面中的重定向装置的功能进行理解。该种可能的实现方式，通过第二处理器中的重定向装置可以从第二VM的虚拟化特权级直接切换到host的用户态，进一步的减少了异步处理过程中的切换开销，进一步的提高了计算机设备的性能。

在第三方面的一种可能的实现方式中，重定向装置还被配置为：将中断处理函数的入口地址写入到第一寄存器中；第二处理器，具体用于从第一寄存器中读取中断处理函数的入口地址，并根据入口地址调用中断处理函数，在host的用户态下执行中断处理函数。

该过程可以通过上述第一方面中的相应内容进行立即，此处不再重复赘述。

在第三方面的一种可能的实现方式中，第二处理器具体用于：从第二VM的上下文切换为影子线程的上下文，并在所述host的用户态下处理中断；在处理完中断后，从影子线程的上下文切换回第二VM的上下文；重定向装置还被配置为：从host的用户态切换回第二VM的虚拟化特权级；第二处理器还用于：继续运行第二VM。

该种可能的实现方式中，执行中断处理函数包括上下文切换的过程和中断处理的过程，该方案通过重定向装置直接从host的用户态切换到第二VM的虚拟化特权级，可以快速启动被中断的第二VM。

在第三方面的一种可能的实现方式中，第二处理器还用于：从第一线程的上下文切换为影子线程的上下文，并在host的用户态下处理中断；在处理完中断后，从影子线程的上下文切换为第一线程的上下文；继续执行第一线程。

该种可能的实现方式中，若第二处理器上没运行虚拟机，则可以直接切换第一线程的上下文和影子线程的上下文，然后处理中断，再切换回第一线程的上下文，进而快速启动第一线程继续运行。

在第三方面的一种可能的实现方式中，host的特权级为内核态， 第二处理器用于：接收中断控制器发送的host的内核态的第二中断请求；跳转到host的内核态的中断处理函数的入口地址，该中断处理函数与第二中断请求对应；在host的内核态下执行所述内核态的所述中断处理函数。

该种可能的实现方式中，若host的特权级为内核态，则中断控制器可以直接向第二处理器发送内核态的中断，第二处理器执行内核态的处理流程来完成相应的操作，在跳转到host的内核态的中断处理函数的入口地址之前，若第二处理器上运行着第二VM，则先从第二VM的虚拟化特权级切换到host的内核态，若第二处理器上没运行虚拟机，host处于用户态，则先从host的用户态切换到host的内核态，若host处于内核态，则直接跳转到中断处

理函数的入口地址，然后执行该中断处理函数。

本申请第四方面提供一种计算机设备，该计算机设备包括：第一处理器、第二处理器和中断控制器；所述计算机设备上运行有宿主机（host），第二处理器还包括硬件辅助切换装置。该第一处理器可以通过访问中断控制器中的用户态地址映射空间或内核态地址映射空间向中断控制器发送中断请求；中断控制器设置影子线程的信息；中断控制器向第二处理器发送所述中断请求。硬件辅助切换装置被配置为：从第一线程切换至影子线程，该第一线程为第二处理器切换前运行的线程，影子线程是运行在用户态的host中的用于包含中断处理函数的线程，中断处理函数与第二中断请求对应；第二处理器具体用于，在host的用户态下执行中断处理函数。

该种可能的实现方式中，硬件辅助切换装置可以是通过硬件电路来实现的。切换线程可以包括切换线程的信息，该第一线程的信息包括该第一线程的内存地址空间，页表首地址、数据结构信息，处理特权级等影子线程是运行在用户态的host中的线程，用于执行中断处理函数。影子线程的信息包括影子线程的内存地址空间，页表首地址、数据结构信息，处理特权级等。第一线程为第二处理器正在运行的线程。影子线程可以理解为是第一线程的影子，第二处理器正在运行的线程被中断后，第二处理器运行影子线程来执行中断处理。该种可能的实现方式中，不需要陷出到host的内核态，通过硬件辅助切换装置可以快速切换第一线程的信息和影子线程的信息，从而可以直接进入host的用户态处理该中断，进一步减少了切换开销，提高了计算机设备的性能。

在第四方面的一种可能的实现方式中，第二处理器还包括重定向装置，第二处理器，还用于在host的用户态下执行中断处理函数之前，确定第一线程是否处于第二VM的虚拟化特权级，第二VM运行在第二处理器上；重定向装置被配置为：若第一线程处于VM的虚拟化特权级，则将VM的虚拟化特权级切换为host的用户态。

该种可能的实现方式中，第二处理器中的重定向装置可以是通过硬件电路实现的，该重定向装置的功能可以参阅上述第一方面中的重定向装置的功能进行理解。该种可能的实现方式，通过第二处理器中的重定向装置可以从VM的虚拟化特权级直接切换到host的用户态，进一步的减少了异步处理过程中的切换开销，进一步的提高了计算机设备的性能。

在第四方面的一种可能的实现方式中，重定向装置还被配置为：将中断处理函数的入口地址写入到第一寄存器中；第二处理器，具体用于从第一寄存器中读取中断处理函数的入口地址，并根据入口地址调用中断处理函数，在host的用户态下执行中断处理函数。

该过程可以通过上述第一方面中的相应内容进行立即，此处不再重复赘述。

在第四方面的一种可能的实现方式中，第二处理器具体用于：从第二VM的上下文切换为影子线程的上下文，并在所述host的用户态下处理中断；在处理完中断后，从影子线程的上下文切换回第二VM的上下文；重定向装置还被配置为：从host的用户态切换回第二VM的虚拟化特权级；第二处理器还用于：继续运行第二VM。

该种可能的实现方式中，执行中断处理函数包括上下文切换的过程和中断处理的过程，该方案通过重定向装置直接从host的用户态切换到第二VM的虚拟化特权级，可以快速启动被中断的第二VM。

在第四方面的一种可能的实现方式中，第二处理器还用于：从第一线程的上下文切换为影子线程的上下文,并在host的用户态下处理中断；在处理完中断后，从影子线程的上下文切换为第一线程的上下文；继续执行第一线程。

该种可能的实现方式中，若第二处理器上没运行虚拟机，则可以直接切换第一线程的上下文和影子线程的上下文，然后处理中断，再切换回第一线程的上下文，进而快速启动第一线程继续运行。

在第四方面的一种可能的实现方式中，host的特权级为内核态， 第二处理器用于：接收中断控制器发送的host的内核态的第二中断请求；跳转到host的内核态的中断处理函数的入口地址，该中断处理函数与第二中断请求对应；在host的内核态下执行所述内核态的所述中断处理函数。

该种可能的实现方式中，若host的特权级为内核态，则中断控制器可以直接向第二处理器发送内核态的中断，第二处理器执行内核态的处理流程来完成相应的操作，在跳转到host的内核态的中断处理函数的入口地址之前，若第二处理器上运行着第二VM，则先从第二VM的虚拟化特权级切换到host的内核态，若第二处理器上没运行虚拟机，host处于用户态，则先从host的用户态切换到host的内核态，若host处于内核态，则直接跳转到中断处理函数的入口地址，然后执行该中断处理函数。

本申请第五方面提供一种异常处理的方法，该方法应用于计算机设备，计算机设备上运行有宿主机host和虚拟机VM，该方法包括：获取VM触发的异常指令；从VM的虚拟化特权级切换到host的用户态；在host的用户态下执异常指令对应的异常处理函数。

在第五方面的一种可能的实现方式中，该方法还包括：将异常处理函数的入口地址写入到第一寄存器中;第一处理器,具体用于从第一寄存器中读取异常处理函数的入口地址，并根据入口地址调用异常处理函数，在host的用户态下执行异常指令对应的异常处理函数。

在第五方面的一种可能的实现方式中，异常指令是VM执行输入/输出I/O操作触发的。

在第五方面的一种可能的实现方式中，该方法还包括：从VM的上下文切换为用户态的影子执行环境的上下文，并在host的用户态下处理异常；所述异常处理函数包含于运行在host的用户态的虚拟处理器的线程中，所述虚拟处理器的线程被分配的执行环境包括主线程执行环境、所述VM的执行环境和影子执行环境，所述虚拟处理器的线程被分配的执行环境为所述虚拟处理器的线程运行所需的资源，所述影子执行环境具有独立于所述主线程执行环境和所述VM的执行环境的用户态内存空间；所述影子执行环境用于执行所述异常处理函数，所述异常处理函数用于处理触发所述异常指令的异常。

该第五方面以及任一可能的实现方式提供的异常处理的方法可以参阅上述第一方面以及第一方面中任一可能的实现方式的相关介绍进行理解，此处不再重复赘述。

本申请第六方面提供中断处理的方法，该方法应用于计算机设备，该计算机设备上运行有宿主机host和第一VM；该方法包括：获取第一VM触发的第一中断请求，该第一中断请求用于请求执行中断处理函数；响应于第一中断请求，确定中断信息；根据中断信息，生成第二中断请求，该第二中断请求用于请求执行所述中断处理函数，响应于所述第二中断请求，执行所述中断处理函数，第二中断请求是通过第一中断请求得到的。

在第六方面的一种可能的实现方式中，第一中断请求包括虚拟事件中断请求号VEI ID，VEI ID用于标识触发第一中断请求的虚拟事件；该方法还包括：获取第一VM的VM ID，该中断信息包括VEI ID和第一VM的VM ID；根据VEI ID和第一VM的VM ID确定VEI物理号，以及处理第一中断请求的host的特权级，VEI物理号用于在host中标识虚拟事件；上述步骤：响应于第二中断请求，执行中断处理函数，包括：获取VEI物理号，根据host的特权级对应的处理方式，响应于接收到第二中断请求，执行与VEI物理号对应的中断处理函数。

在第六方面的一种可能的实现方式中，第一中断请求包括虚拟事件中断请求号VEI ID，VEI ID用于标识触发第一中断请求的虚拟事件；中断信息包括VEI ID；该方法还包括：根据VEI ID确定处理第一中断请求的host的特权级；上述步骤：响应于第二中断请求，执行中断处理函数，包括：获取VEI ID，根据host的特权级对应的处理方式，响应于接收到第二中断请求，执行与VEI ID对应的中断处理函数。

在第六方面的一种可能的实现方式中，host的特权级为用户态；上述步骤：响应于第二中断请求，执行中断处理函数，包括：从第一线程切换至影子线程，第一线程为宿主机切换前运行的线程，影子线程是运行在用户态的host中的线程，影子线程用于执行中断处理函数；在host的用户态下执行中断处理函数。

在第六方面的一种可能的实现方式中，该方法还包括：在host的用户态下执行中断处理函数之前，确定第一线程是否处于第二VM的虚拟化特权级，第二VM运行在计算机设备上；若被中断的线程处于第二VM的虚拟化特权级，则将第二VM的虚拟化特权级切换为host的用户态。

在第六方面的一种可能的实现方式中，该方法还包括：将中断处理函数的入口地址写入到第一寄存器中；从第一寄存器中读取中断处理函数的入口地址，并根据入口地址调用中断处理函数，在host的用户态下执行中断处理函数。

在第六方面的一种可能的实现方式中，上述步骤：在所述host的用户态下执行所述中断处理函数，包括：从第二VM的上下文切换为影子线程的上下文，并在host的用户态下处理中断；在处理完中断后，从影子线程的上下文切换回第二VM的上下文；从host的用户态切换回第二VM的虚拟化特权级；继续运行第二VM。

在第六方面的一种可能的实现方式中，该方法还包括：从第一线程的上下文切换为影子线程的上下文，并在host的用户态下处理中断；在处理完中断后，从影子线程的上下文切换为第一线程的上下文；继续执行第一线程。

在第六方面的一种可能的实现方式中，host的特权级为内核态，上述步骤：根据host的特权级对应的处理方式，处理第二中断请求，以执行与VEI物理号对应的操作，包括：跳转到host的内核态的中断处理函数的入口地址，中断处理函数与第二中断请求对应；在host的内核态下执行内核态的中断处理函数。

该第六方面以及任一可能的实现方式提供的异常处理的方法可以参阅上述第三方面以及第三方面中任一可能的实现方式的相关介绍进行理解，此处不再重复赘述。

本申请第七方面提供一种计算机设备，该计算机设备具有实现上述第五方面或第五方面任意一种可能实现方式的方法的功能。该功能可以通过硬件实现，也可以通过硬件执行

相应的软件实现。该硬件或软件包括一个或多个与上述功能相对应的模块，例如：接收单元和处理单元。

本申请第八方面提供一种计算机设备，该计算机设备具有实现上述第六方面或第六方面任意一种可能实现方式的方法的功能。该功能可以通过硬件实现，也可以通过硬件执行相应的软件实现。该硬件或软件包括一个或多个与上述功能相对应的模块，例如：接收单元和处理单元。

本申请第九方面提供一种存储一个或多个计算机执行指令的计算机可读存储介质，当计算机执行指令被处理器执行时，处理器执行如上述第五方面或第五方面任意一种可能的实现方式的方法。

本申请第十方面提供一种存储一个或多个计算机执行指令的计算机可读存储介质，当计算机执行指令被处理器执行时，处理器执行如上述第六方面或第六方面任意一种可能的实现方式的方法。

本申请第十一方面提供一种存储一个或多个计算机执行指令的计算机程序产品，当计算机执行指令被处理器执行时，处理器执行如上述第五方面或第五方面任意一种可能的实现方式的方法。

本申请第十二方面提供一种存储一个或多个计算机执行指令的计算机程序产品，当计算机执行指令被处理器执行时，处理器执行如上述第六方面或第六方面任意一种可能的实现方式的方法。

本申请第十三方面提供了一种芯片系统，该芯片系统包括至少一个处理器，至少一个处理器用于支持进程间通信的装置实现上述第五方面或第五方面任意一种可能的实现方式中所涉及的功能。在一种可能的设计中，芯片系统还可以包括存储器，存储器，用于保存消息管理装置必要的程序指令和数据。该芯片系统，可以由芯片构成，也可以包含芯片和其他分立器件。

本申请第十四方面提供了一种芯片系统，该芯片系统包括至少一个处理器，至少一个处理器用于支持进程间通信的装置实现上述第六方面或第六方面任意一种可能的实现方式中所涉及的功能。在一种可能的设计中，芯片系统还可以包括存储器，存储器，用于保存调度装置必要的程序指令和数据。该芯片系统，可以由芯片构成，也可以包含芯片和其他分立器件。

本申请实施例通过重定向装置来实现同步处理过程中的特权级切换，通过虚拟事件通知装置和中断控制器不经过 host 的内核态就可以实现异步的中断请求传递，无论是同步处理过程还是异步处理过程都减少了切换开销，提高了计算机设备的性能。

附图说明

图1是本申请实施例提供的计算机设备的一结构示意图；

图2是本申请实施例提供的计算机设备的另一结构示意图；

图3是本申请实施例提供的异常处理的方法的一实施例示意图；

图4是本申请实施例提供的处理器的一结构示意图；

图5是本申请实施例提供的虚拟处理器的线程的一结构示意图；

图6是本申请实施例提供的计算机设备的另一结构示意图；
图7A是本申请实施例提供的计算机设备的另一结构示意图；
图7B是本申请实施例提供的中断状态的变化示意图；
图8是本申请实施例提供的计算机设备的另一结构示意图；
图9是本申请实施例提供的影子线程的一结构示意图；
图10是本申请实施例提供的中断处理的方法的一实施例示意图；
图11是本申请实施例提供的计算机设备的另一结构示意图；
图12是本申请实施例提供的计算机设备的另一结构示意图；
图 13 是本申请实施例提供的计算机设备的另一结构示意图。

具体实施方式

下面结合附图，对本申请的实施例进行描述，显然，所描述的实施例仅仅是本申请一部分的实施例，而不是全部的实施例。本领域普通技术人员可知，随着技术的发展和新场景的出现，本申请实施例提供的技术方案对于类似的技术问题，同样适用。

本申请的说明书和权利要求书及上述附图中的术语“第一”、“第二”等是用于区别类似的对象，而不必用于描述特定的顺序或先后次序。应该理解这样使用的数据在适当情况下可以互换，以便这里描述的实施例能够以除了在这里图示或描述的内容以外的顺序实施。此外，术语“包括”和“具有”以及他们的任何变形，意图在于覆盖不排他的包含，例如，包含了一系列步骤或单元的过程、方法、系统、产品或设备不必限于清楚地列出的那些步骤或单元，而是可包括没有清楚地列出的或对于这些过程、方法、产品或设备固有的其它步骤或单元。

本申请实施例提供一种计算机设备、异常处理的方法以及中断处理的方法，用于减少切换开销，提高计算机设备的性能。本申请还提供了相应的装置、计算机可读存储介质、计算机程序产品等。以下分别进行详细说明。

虚拟化是将计算机设备的硬件层中的硬件资源（例如处理器、存储器中的存储空间以及网络资源）虚拟化后共享给多个虚拟计算机使用。虚拟计算机为所有类型的虚拟化设备中通过软件虚拟出来的运行环境的统称，该概念包括虚拟机或容器。

如图1所示，计算机设备100包括硬件层112、宿主机层109和虚拟化层，虚拟化层包含虚拟机101和102。虚拟机的个数还可以更多或更少，在此只以两个为例。硬件层112包括处理器系统114、存储器113、通信接口115和中断控制器116。

虚拟机（virtual machine，VM）是通过虚拟化软件在计算机设备上模拟出的一台或者多台虚拟计算机。这些虚拟机运行在完全隔离的环境中，就像真正的计算机那样进行工作。虚拟机（图1中101和102）上可以安装客户操作系统（guest operating system, guest OS）（图1中105和106），客户操作系统上运行有一个或多个应用程序（图1中103和104）。虚拟机还可访问网络资源。对于在虚拟机中运行的应用程序而言，就像是在真正的计算机中工作。

VM在运行时会涉及到虚拟机特权级，该虚拟机特权级包括虚拟机的用户态和虚拟机的内核态。虚拟机的内核态和虚拟机的用户态的划分是将虚拟机当做真正的计算机来划分的。

在介绍用户态和内核态的概念之前先介绍下用户空间和内核空间。

现有的某些计算机系统中，将整个虚拟地址空间进行划分，划分出两块虚拟地址空间分别作为内核空间（内核态虚拟地址空间）和用户空间（用户态虚拟地址空间）。以64位体系架构下，48位有效虚拟地址为例，用户空间为(00000000,00000000)-(0000FFFF,FFFFFFFF)，内核空间为(FFFF0000,00000000)-(FFFFFFFF,FFFFFFFF)。内核空间和用户空间分别有对应的寄存器，寄存器中存有对应空间的独立的页表基地址。内核空间还可以称为内核态内存空间，用户空间还可以称为用户态内存空间。而运行在用户空间的进程或线程可称为用户态的进程或线程，运行在内核空间的进程或线程称为内核态的进程或线程。

虚拟处理器（如图1中107和108）：在虚拟化技术下，代表以共享或者分片方式提供给虚拟计算机使用的物理处理单元，例如虚拟CPU(virtual central processing unit，vCPU)。一台虚拟计算机可以有一个或多个虚拟处理器为其服务，当存在多个虚拟处理器时，通常有一个虚拟处理器为主虚拟处理器，其他为从虚拟处理器。虚拟机包含的虚拟存储器等其他虚拟硬件资源在图1中未示出。应理解，虚拟机相当于一台独立的计算机，所以虚拟机执行动作也可以认为是虚拟处理器执行该动作。虚拟处理器是通过虚拟化软件虚拟出的，它的运行实际是宿主机的处理器或物理核读取并运行软件程序实现的，例如一个物理核读取软件程序并在该物理核的硬件辅助虚拟化的特定模式（例如x86的non-Root模式）下运行该软件程序以实现一个虚拟处理器。一台虚拟机的多个虚拟处理器可以位于不同的物理核上。

虚拟处理器陷入（trap in）和虚拟处理器陷出（trap out）：虚拟化系统包括两种模式：宿主模式（host mode）与客户模式（guest mode）。宿主模式也可以称为是host的特权级，如：host的用户态或host的内核态，客户模式也可以称为是VM的特权级，如：VM的用户态或VM的内核态。当一个虚拟处理器进入客户模式，叫做陷入（虚拟）；当该虚拟处理器离开客户模式，叫陷出（虚拟）。虚拟处理器陷出后物理处理器将暂时不执行该虚拟处理器的代码，所以此时可以理解为虚拟处理器没有运行。针对一个物理处理器而言，其上运行的虚拟处理器陷入，则可以认为该物理处理器处于客户模式，运行虚拟处理器的代码，当其上运行的虚拟处理器陷出到宿主模式，则可以认为该物理处理器处于宿主模式，运行宿主机相关的代码，比如虚拟机监视器。

宿主机（host）层109作为管理层，用以完成硬件资源的管理、分配，为虚拟机呈现虚拟硬件平台，实现虚拟机的调度和隔离等。在一些实现方式下，宿主机层109可以包括宿主机操作系统111和虚拟监控装置，例如虚拟机监视器110(virtual machine monitor, VMM)。其中虚拟监控器110可部署在宿主机操作系统111之内，也可以部署在宿主机操作系统111之外。在其他虚拟化架构中虚拟监控装置还可以称为hypervisor或其他类型的虚拟监控装置。在另一些实现方式下，例如在虚拟化架构Xen中，宿主机层109还可以包括1个特权虚拟机。其中，虚拟硬件平台对其上运行的各个虚拟计算机提供各种硬件资源，如虚拟处理器、虚拟内存、虚拟磁盘、虚拟网卡等。虚拟计算机则运行在宿主机层为其准备的虚拟硬件平台上。宿主机层109还可以称为虚拟化平台，有时宿主机层还可以简称为宿主机。宿主机的特权级包括用户态和内核态。

硬件层112：虚拟化环境运行的硬件平台。其中，硬件层可包括多种硬件，如图1所示，硬件层可包括处理器系统114和存储器113，还可以包括通信接口115，例如网卡（network interface card，NIC）；还可以包括中断控制器116、输入/输出（input/output ， I/O）设备等。其中处理器系统114可以包括一个或多个处理器，如图1中列举的处理器1和处理器2。每个处理器中可包括多个物理核，处理器中还可以包括多个寄存器，如：通用寄存器、浮点寄存器等。

处理器系统114，有时称为物理处理器。物理核代表处理器中最小处理单元，各个处理器包含的核的个数可以相同，也可以不同。具有一个核的处理器称为单核处理，具有多个物理核的处理器被称为多核处理器。按照内核架构是否相同，可以分为同构多核与异构多核。虚拟处理器和物理核可以是绑定的关系，即一个虚拟处理器固定在某个物理核上运行，不能被调度到其他物理核上运行，则该虚拟处理器为绑核；一个虚拟处理器可以根据需要被调度到不同的物理核上运行，则该虚拟处理器为非绑核。

中断控制器116：设置在触发中断请求的硬件和处理器之间，主要用于收集各个硬件产生的中断请求，并按照一定的优先级或其他规则发送给处理器。例如高级可编程中断控制器(advanced programmable interrupt controller, APIC)。

中断（interruption）是指暂停当前程序的指令转而执行中断服务程序。

中断服务程序（interrupt service routine, ISR），还可称为中断处理函数，是用于处理中断请求的程序。当处理器接收到中断请求时，暂时停止当前程序的执行转而执行该中断请求对应的中断服务程序。

中断请求（interrupt request）：中断请求指的是硬件产生的一种事件，硬件将该事件发送到处理器，当处理器接收到该事件时，暂时停止当前程序的执行转而执行该事件对应的程序。硬件产生中断请求可能是硬件自己触发的，也可能是软件触发硬件产生的。中断请求有时也简称为中断。计算机中的某些硬件（例如网卡、声卡、鼠标、硬盘等）能在没有处理器介入的情况下完成一定的工作，但是这些硬件还是需要通过中断控制器，让处理器为其做一些特定的工作。中断号为一个中断请求的标识，本申请中用英文IRQ ID表示。

硬件层112还可以包括内存管理单元（memory management unit，MMU）。MMU是一种负责处理内存访问请求的计算机硬件。它的功能包括虚拟地址到物理地址的转换（即虚拟内存管理）、内存保护、中央处理器高速缓存的控制。MMU通常借助转译后备缓冲器（translation lookaside buffer， TLB）的相联高速缓存（associative cache）来将虚拟页号转换为物理页号。

存储器113提供的存储空间（地址空间）被划分给虚拟机和宿主机使用。宿主机物理地址 (host physical address，HPA)指的是本地主机（宿主机）可以使用的物理地址空间；宿主机虚拟地址 (host virtual address，HVA)是本地主机（宿主机）可使用的虚拟地址空间。客户机物理地址 (guest physical address，GPA)是虚拟机的客户操作系统可使用的物理地址空间；客户机虚拟地址 (guest virtual address，GVA)是虚拟机的客户操作系统可使用的虚拟地址空间。

计算机设备100可以为物理设备，例如服务器或终端设备。终端设备可以是具有无线连

接功能的手持式设备、或连接到无线调制解调器的其他处理设备。例如，可以为移动电话、计算机(personal computer,PC)、平板电脑、个人数码助理(personal digital assistant,PDA)、移动互联网设备(mobile Internet device,MID)、可穿戴设备和电子书阅读器(e-book reader）等；也可以是便携式、袖珍式、手持式、计算机内置的或者车载的移动设备。

上述计算机设备100中的VM在执行一些操作（如：对串口的内存的读/写操作，对磁盘的内存的读/写操作，对网卡的内存的读/写操作、缺页异常的操作等）时，会触发异常或中断（异常通常也描述为同步异常，中断通常也描述为异步异常），硬件层或宿主机的内核会针对该异常或中断进行处理。本申请实施例中对异常可以采用同步处理的方式，对中断可以采用异步处理的方式。同步处理指的是在同一个处理器上的处理，异步处理指的是跨处理的处理。在虚拟I/O的场景，针对虚拟I/O的处理可分为同步处理和异步处理两种。当该VM执行所涉及到的I/O设备是串口、显卡、字符或USB等设备时，一般采用同步处理方式，当VM执行所涉及到的I/O设备是磁盘或网卡等设备时，通常采用异步处理方式。

本申请后文实施例中使用了第一处理器和第二处理器，需要说明的是，第一处理器和第二处理器也可以是同一个多核处理器中的两个不同的核。

下面从同步处理过程和异步处理的过程分别进行介绍。

一、同步处理过程。

如图2所示，计算机设备包括第一处理器、该计算机设备上运行有宿主机（host）和第一VM，具体的，第一VM运行在第一处理器上。第一VM在执行一些操作时时会触发异常指令，该异常指令可以是VM执行输入/输出（input/output,I/O）操作时触发的，该I/O操作可以是VM要访问虚拟I/O设备触发的操作，通常，虚拟I/O设备包括前端驱动和后端设备，前端驱动运行在VM中，后端设备的程序运行在host中，若VM执行的操作要通过前端驱动访问后端设备，则会触发异常指令，通过该异常指令实现对后端设备的访问。虚拟I/O设备是后端设备的一种，访问虚拟I/O设备可以是要从虚拟I/O设备的内存中读取数据或者向该虚拟I/O设备的内存写入数据，也可能是对虚拟I/O设备的其他控制操作。虚拟I/O设备是虚拟机能使用的I/O设备，该虚拟I/O设备可以是通过软件模拟出来的纯虚拟化的I/O设备，该虚拟I/O设备也可以通过映射的方式映射到物理I/O设备上，访问该虚拟I/O设备，最终会访问到物理I/O设备。第一处理器接收该异常指令，若第一处理器确定该异常指令需要在host的用户态处理；则会从第一VM的虚拟化特权级切换到host的用户态，在host的用户态下执行该异常处理函数。以虚拟I/O设备包括串口、显卡、字符或USB等虚拟为例，串口、显卡、字符或USB都会有各自的地址范围，第一处理器可以根据该地址范围确定出本次I/O操作要操作的虚拟I/O设备。本申请中同步执行与虚拟I/O设备相关的操作，若该虚拟I/O设备是通过软件模拟出来的虚拟I/O设备，例如：liunx系统的虚拟串口。通过软件模拟出来的虚拟I/O设备可以不对应的物理I/O设备。虚拟机中的虚拟I/O设备也可以关联物理I/O设备，如：虚拟鼠标关联物理鼠标，当虚拟I/O设备关联物理I/O设备时，第一处理器运行该用户态的线程通过系统调用陷入内核态，然后调用物理I/O设备的驱动，从而操作物理I/O设备。

第一VM触发的异常指令可以有多种类型，如：非法指令异常、缺页异常、内存访问异常、地址对齐异常、访问虚拟I/O设备引起的异常等。第一处理器可以根据异常指令的类型

确定该异常指令需要在host的用户态处理或者host的内核态处理。针对需要在host的用户态处理的异常指令，就可以执行上述重定向操作。

该第一处理器可以包括第一重定向装置，第一重定向装置可以执行从第一VM的虚拟化特权级到host的用户态的特权级切换，以及将异常处理函数的入口地址写入到第一寄存器中的过程。这样，第一处理器就可以从第一寄存器中读取异常处理函数的入口地址，并根据入口地址调用异常处理函数，在host的用户态下执行异常指令对应的异常处理函数。该第一重定向装置的功能可以是通过硬件电路实现的。

用户态的宿主机可以包括虚拟处理器（virtual central processing unit，vCPU）的线程，该线程中可以包括影子执行环境。第一处理器可以在影子执行环境中执行异常处理函数。

下面结合图3介绍通过第一处理器、第一重定向装置和影子执行环境来实现上述同步处理的异常处理的方法的过程。

201、第一处理器获取第一VM触发的异常指令。

该步骤201的过程可以是第一VM在执行超出第一VM权限的操作时发生了异常，第一处理器可以通过硬件逻辑检测到该异常代码，进而获取到该异常指令。

第一处理器在获取异常指令后，可以将该异常产生的原因、异常指令的类型，第一VM所要访问的内存，以及返回地址都写入到硬件寄存器中，供后续处理异常使用。

可选的，202、第一处理器确定该异常指令需要在host的用户态处理。

第一处理器可以根据该异常指令所对应的虚拟I/O设备的地址范围来确定该异常指令需要再host的用户态处理。

若该异常指令需要在host的内核态处理，则第一重定向装置从第一VM的虚拟化特权级切换到host的内核态，然后跳转到host的内核态的异常处理函数的入口进行处理即可。

该步骤202为可选步骤，也可以不执行该步骤202。若不执行步骤202，则可以直接执行步骤203。

203、第一重定向装置从第一VM的虚拟化特权级切换为host的用户态。

VM的虚拟化特权级表示处于虚拟化模式下的特权级，也就是VM的工作模式。该虚拟化特权级包括虚拟化的用户态和虚拟化的内核态。从VM的虚拟化特权级切换（直接切换）到host的用户态可以包括将VM的虚拟化特权级的信息从用于存储特权级的寄存器中移出，然后将host的用户态的信息写入到该寄存器的过程，该寄存器可以是系统寄存器。本申请中特权级和虚拟化可以分别用1个比特位来表示，例如：用0表示用户态，用1来表示内核态，用0来表示非虚拟化，用1来表示虚拟化。特权级在第一个比特位，是否虚拟化在第二个比特位，例如：00表示非虚拟化的用户态，也可以称为宿主机的用户态，10示非虚拟化的内核态，也可以称为宿主机的内核态，01表示虚拟化的用户态，也可以称为VM的用户态，11表示虚拟化的内核态，也可以称为VM的内核态。当然，本处指示列举了几个示例，关于特权级与虚拟化的顺序，以及可能的其他的表现形式本申请中不做限定。

204、第一重定向装置将异常处理函数的入口地址写入到第一寄存器中。

205、第一处理器在影子执行环境中，将第一VM的上下文切换为用户态的影子执行环境

的上下文，并执行异常处理。

影子执行环境是用户态的异常处理函数的载体，该异常处理函数也可以称为处理异常中断的服务例程。

该异常处理函数包含于运行在host的用户态的虚拟处理器的线程中，虚拟处理器的线程被分配的执行环境包括主线程执行环境、VM的执行环境和影子执行环境，虚拟处理器的线程被分配的执行环境为虚拟处理器的线程运行所需的资源，影子执行环境具有独立于主线程执行环境和VM的执行环境的用户态内存空间；影子执行环境用于执行异常处理函数，异常处理函数用于处理触发异常指令的异常。

可选的，206、在处理异常的过程中，判断是否产生需要在内核处理的中断或异常，若产生需要在内核处理的中断或异常，则执行207，若未产生需要在内核处理的中断或异常，则执行209。

207、若产生需要在内核处理的中断或异常，则将用户态的影子执行环境的上下文切换为内核态的影子执行环境的上下文，并在内核态处理该中断或异常。

208、需要在内核处理的中断或异常处理完成后，将内核态的影子执行环境的上下文切换为用户态的影子执行环境的上下文。

209、继续执行异常中断处理。

210、在执行完异常中断处理后，将用户态的影子执行环境的上下文切换为第一VM的上下文。

211、第一重定向装置从host的用户态切换回第一VM的虚拟化特权级。

该切换过程为将host的用户态的信息从特权寄存器中转移出，将原来存储在与该特权寄存器对应的内存或寄存器中的第一VM的虚拟化特权级的信息再写入到该特权寄存器中。

212、第一重定向装置将执行流程跳转到第一VM中被打断的指令，第一VM继续执行该被打断的指令。

以上描述了同步处理的过程，该同步处理过程中，如图4所示，第一重定向装置可以包括：特权级选择模块和特权级重定向模块。特权级重定向模块还可以包括更新模块和还原模块。

其中，特权级选择模块可以用于确定第一VM运行期间触发的异常指令是否需要重定向到host的用户态进行处理。

如果需要重定向到host的用户态，则特权级选择模块通知特权级重定向模块进行重定向操作。重定向的过程可以参阅上述步骤204的描述进行理解。

更新模块用于：

（1）记录切换信息：切换信息包括切换前的第一VM的虚拟化特权级的信息和切换之后的host的用户态的信息。

（2）更新状态控制信息：将状态寄存器的信息更新为用户态的异常处理函数的相关信息，该相关信息包括但不限于异常处理函数的入口地址，异常产生的原因，产生异常的内存地址或指令等。

（3）切换特权级：将第一VM的虚拟化特权级切换为host的用户态，并跳转到host的用

户态的异常处理函数的入口地址。

还原模块用于：根据记录的第一VM的虚拟化特权级的信息，从host的用户态切换回第一VM的虚拟化特权级，并根据返回地址跳转到第一VM被打断的指令，使得第一VM继续执行该指令。

上述影子执行环境包括用户态的影子执行环境和内核态的影子执行环境，即该影子执行环境在用户态和内核态都有独立的可保存上下文信息的存储空间，该存储空间可以是栈，用户态的存储空间可以是用户态的栈，内核态的存储空间可以是内核态的栈。

影子执行环境位于虚拟处理器的线程的执行环境中，该虚拟处理器的线程的任务结构可以参阅图5进行理解。

如图5所示，虚拟处理器的线程的任务可以包括：模拟虚拟处理器的任务、运行VM的任务，以及异常/中断处理的任务。

其中，模拟虚拟处理器的任务会用到vCPU的用户态的主线程执行环境，以及vCPU的内核态的主线程执行环境，该vCPU的内核态的主线程执行环境包括第一保存装置和第二保存装置，该第一保存装置用于保存用户态上下文或内核态上下文，第二保存装置用于保存host上下文或与VM上下文。

运行VM的任务会用到VM执行环境，该VM执行环境用于运行VM。

中断处理的任务会用到影子执行环境，该影子执行环境包括用户态的影子执行环境，以及内核态的影子执行环境。该用户态的影子执行环境包括第一切换装置，内核态的影子执行环境第二切换装置。

其中，第一切换装置用于从VM上下文切换到host的用户态上下文，如上述步骤205和210。该第一切换装置保存VM的运行状态，并恢复影子执行环境的用户态的运行状态，该运行状态的信息包括但不限于通用寄存器、浮点寄存器以及系统寄存器等寄存器中的信息。

第二切换装置用于从host的用户态上下文切换到host的内核态上下文，如上述步骤207和208。该第二切换装置用于保存host的用户态上下文，并恢复内核态上下文，进入内核态，处理发生在影子执行环境的用户态运行期间发生的需要在内核态处理的中断或异常。需要在内核态处理的中断或异常处理完成后，该装置恢复用户态上下文，切换回用户态的影子执行环境继续执行异常指令。内核态的影子执行环境的上下文与虚拟处理器的线程的内核态上下文互不冲突，互不影响。

上述用户态与内核态上下文保存装置、host与VM上下文保存装置、VM与用户态的影子执行环境的上下文切换装置、用户态与内核态上下文切换装置可以是通用寄存器、浮点寄存器以及系统寄存器等。

本申请中的“执行环境”包括软件执行环境和硬件执行环境两个方面，软件执行环境包括虚拟处理器的线程运行在不同执行环境时所使用的栈中的信息，以及线程的上下文的信息，硬件方面包括线程运行时所使用的硬件资源，包括内存资源或处理资源等。

本申请实施例提供的计算机系统，通过同步处理的方式将VM运行期间所发生的异常重定向到host的用户态进行处理，避免了host的内核与用户态之间的切换开销，提高了计算机系统的性能。

二、异步处理。

异步处理指的是跨处理器的处理过程，即运行在第一处理器上的第一VM因执行需要访问第一VM的后端设备的操作而触发的中断请求，需要通过第二处理器才能完成处理。因此，异步处理的过程可以划分为中断请求发送端的流程和中断请求接收端的流程。发送端的流程可以不经过host内核，也可以经过host内核，接收端的流程可以不经过host内核，也可以经过host内核。异步处理的方式可以有如下四种。

异步处理方式1：发送端的流程不经过host内核，接收端的流程也不经过host内核。

如图6所示，该计算机系统包括第一处理器、第二处理器、中断控制器和物理I/O设，该计算机设备上运行有宿主机host，第一虚拟机VM运行在第一处理器上。该计算机系统还包括虚拟事件中断组件，该虚拟事件中断组件可以包括虚拟事件通知装置和虚拟事件中断装置，虚拟事件通知装置电路结构位于第一处理器中，虚拟事件中断装置的电路结构位于中断控制器中。

第一处理器用于：接收第一VM触发的第一中断请求，该第一中断请求用于请求执行中断处理函数；虚拟事件通知装置被配置为：响应于第一中断请求，直接向中断控制器发送中断信息；中断控制器被配置为：根据中断信息确定第二处理器的标识，根据第二处理器的标识向第二处理器发送第二中断请求；第二处理器用于：响应于接收到的第二中断请求，执行所述中断处理函数。

该虚拟事件中断组件的结构可以参阅图7A进行理解。如图7A所示，该虚拟事件中断组件包括虚拟事件生成模块、虚拟事件监测模块、虚拟事件中断映射表、虚拟事件中断查询模块，以及虚拟事件中断信息设置模块。其中，虚拟事件生成模块和虚拟事件监测模块位于第一处理器中，虚拟事件中断映射表、虚拟事件中断查询模块，以及虚拟事件中断信息设置模块位于中断控制器中。

虚拟事件生成模块中可以包括可被第一VM直接访问的VEI-REG寄存器，这样，第一VM访问该虚拟事件生成模块时VM可以将虚拟事件中断请求号（virtual event interrupt identifier，VEI ID）写入该VEI-REG寄存器中。第一VM访问该虚拟事件生成模块的方式包括但不限于读写专用寄存器，执行专用指令，执行现有指令触发专用硬件逻辑等。

该VEI ID用于标识第一VM触发第一中断请求的虚拟事件，通常VM触发的中断类型还可以包括局部中断、外部中断以及时钟中断等。该VEI ID指示该中断请求是因虚拟事件触发的中断，则虚拟事件生成模块通知虚拟事件监测模块开始工作。虚拟事件监测模块查询第一VM的编号，也就是虚拟机标识（virtual machine identifier，VM ID）。虚拟事件监测模块可以从用于存放虚拟机标识的寄存器中获取该VM ID。该VM ID在第一处理器运行第一VM时写入到该用于存放虚拟机标识的寄存器中，该用于存放虚拟机标识的寄存器可以是控制状态寄存器（control status registers， CSRs）。上述中断信息中包括VEI ID和VM ID。

虚拟事件监测模块向虚拟事件中断查询模块发送该VEI ID以及第一VM的虚拟机标识（virtual machine identifier，VM ID）。

本申请实施例中，为了实现第一VM不陷出（trap out）到host，新增了可被VM直接访问的VEI-REG寄存器。VM可以将VEI ID写入VEI-REG寄存器中，从而使第一处理器可以通过

该VEI-REG寄存器读取VEI ID，然后再从CSRs寄存器读取VM ID，进而通过中断控制器和第二处理器异步执行后续的中断处理，第一VM在向VEI-REG寄存器写入VEI ID后，因为后续中断过程通过异步执行，所以第一VM不需要退出，可以继续运行。

虚拟事件中断映射表维护了VEI ID和VMID，与处理器的标识（CPU ID）和VEI物理号，以及该VEI ID和VMID所对应的中断请求所对应的host的特权级（该中断请求应在用户态处理或内核态处理）的映射关系。VEI ID用于指示虚拟机的虚拟事件中断，VEI物理号用于处理器识别虚拟事件中断。VEI ID的取值范围通常为0-63，VEI物理号的取值范围通常为0-2047。本申请实施例中，VEI ID是用于标识虚拟机中发生的虚拟事件，VEI物理号在host中指示虚拟事件的标识。VEI ID与VEI物理号具有映射关系，相互映射的VEI ID与VEI物理号指示同一虚拟事件（如：读/写磁盘的事件，读/写网卡的事件等），VEI ID可以通过数值来表示，如：0至63，每个数值标识一个不同的虚拟事件，一个I/O设备上可以有多个不同的虚拟事件，如：对读磁盘的事件可以用数值1来表示，写磁盘的事件可以用数值2来表示，针对磁盘的某个控制事件可以用数值3来表示，读网卡的事件可以用数值4来表示，写网卡的事件可以用数值5来表示等。每个虚拟机中的同一虚拟事件可以使用相同的VEI ID 。VEI物理号可以是全局唯一的，在不同的虚拟机上发生的同一VEI ID的虚拟事件可以对应不同的VEI物理号。例如：VM1的VEI ID的取值范围对应的VEI物理号可以取0至63，VM2的VEI ID的取值范围对应的VEI物理号可以取64至127，依次类推，这样，第二处理器根据不同的VEI物理号所在的范围就可以确定发生虚拟事件的虚拟机，然后再根据VEI物理号的具体取值就可以确定该VM上所发生的虚拟事件，例如：VEI ID=5时表示读磁盘的事件，那么VEI物理号=5，就表示VM1上发生了读磁盘的事件，若VEI物理号=69，则表示VM2上发生了读磁盘的事件。关于虚拟事件与VEI ID的对应关系，以及VEI物理号与虚拟机的对应关系可以根据需求设置，本申请中对此不做限定。

虚拟事件中断映射表还维护了虚拟事件中断对应的中断处理函数的信息（也可以称为服务例程信息），包括但不限于该服务例程的内存地址空间，数据结构信息，处理特权级等。

虚拟事件中断查询模块通过查找虚拟事件中断映射表，查看VEI ID是否注册或存在。若已注册或已存在，则虚拟事件中断查询模块向虚拟事件中断信息设置模块传递VIE ID和VMID；如果VEI ID未注册或不存在，则忽略该中断请求或抛出异常。

虚拟事件中断信息设置模块基于VEI ID和VMID查询虚拟事件中断映射表，确定本次中断请求对应的第二处理器的CPU ID，以及VEI物理号，以及对应的host的特权级。

本申请中，因为每个虚拟机中的同一虚拟事件可以使用相同的VEI ID，为了区分该虚拟事件所发生的虚拟机，第一处理器可以获取VM ID，VM ID可以是第一处理器从用于存储虚拟机标识的寄存器中读取的。然后虚拟事件通知装置向中断控制器发生VEI ID和VM ID，中断控制器可以通过VEI ID和VM ID查找第二处理器的标识和VEI物理号，以及对应host的特权级。中断控制器中可以维护有输入是：（VEI ID）+（VM ID）->输出是：（CPU ID）+（VEI物理号）+（host的特权级）的映射表。中断控制器可以通过该映射表直接确定出对应的（CPU ID）+（VEI物理号）+（host的特权级）。中断控制器根据CPU ID可以确定接下来执行该异步中断处理的第二处理器，根据该host的特权级可以确定接下来是采用用户态

的处理方式还是内核态的处理方式。第二处理器可以从存储该VEI物理号的寄存器中读取该VEI物理号，然后按照用户态的处理流程或者内核态的处理流程进行相应处理。

上述方式中用到了VM ID和VEI物理号，若VEI ID是全局唯一的，不同的VM针对同一虚拟事件使用不同的VEI ID，这样就不需要再使用VM ID，也不需要再转换VEI 物理号，只要根据该VEI ID确定对应的第二处理器的标识以及host的特权级即可执行后续的中断处理。该种实施例中，上述中断信息中包括VEI ID，不再需要VM ID。

若本次中断请求需要在host的用户态处理，则该虚拟事件中断信息设置模块设置第二处理器中的用户态的中断处理线程的信息。若本次中断请求需要在host的内核态处理，则该虚拟事件中断信息设置模块通知第二处理器处理内核中断。

本申请实施例中，本次中断请求需要在host的用户态处理，则虚拟事件中断信息设置模块设置完成用户态的中断处理线程的信息后，通知第二处理器处理用户态的中断请求。

第二处理器接收到用户态的中断请求后，会进行相应的中断处理。

本申请实施例中，中断控制器中还会通过两个寄存器维护一个表示虚拟事件中断VEI的一个状态机，每个VEI ID对应一个状态机，该状态机中在处理中断的不同阶段可以包括四种状态，分别为如图7B所示的非激活态（inactive）、待处理状态（pending）、激活状态（active）和（激活+待处理）状态（pending and active）。其中激活态也可以理解为正在处理状态，（激活+待处理）状态表示当前的中断还在处理中，又来了一个新的中断。

两个寄存器中的一个寄存器（如：寄存器1）用于存储是否处于激活状态的值，另一个寄存器（如：寄存器2）用于存储是否处于待处理状态的值，也可以理解为通过两个比特位来表示四种状态，例如：第一个比特位表示是否处于待处理状态，可以用1表示处于待处理状态。第二个比特位表示是否处于激活状态，可以用0表示处于非激活状态，用1表示处于激活状态。这样，非激活状态就可以用00表示，待处理状态就可以用10表示，激活状态就可以01表示，（激活+待处理）状态就可以用11表示。

当中断控制器没有获取到第一中断请求时，寄存器1和寄存器2中的状态值都是0，也就是非激活态的状态值00。

S1：中断控制器获取到中断请求1，则需要转到待处理状态，则将寄存器1中的状态值设置为1。

这样寄存器1中的状态值是1，寄存器2中的状态值是0，也就是待处理状态的状态值10。

S2、中断控制器开始处理该1中断请求1，则需要转到激活状态，则将寄存器1中的状态值清除，也就是恢复到0，将寄存器2中的状态值设置为1。

这样寄存器1中的状态值是0，寄存器2中的状态值是1，也就是激活状态的状态值01。

S3、中断控制器在处理中断请求1时，又接收到一个中断请求2，则需要转到（激活+待处理）状态，则将寄存器1中的状态值设置为1。

这样寄存器1中的状态值是1，寄存器2中的状态值是1，也就是待处理状态的状态值11。

S4、若中断控制器处理完中断请求1后，需要转到非激活状态，则需要清除寄存器1中的状态值和寄存器2中的状态值。

寄存器1和寄存器2中的状态值都是0，也就是非激活态的状态值00。

S5、若中断控制器在（激活+待处理）状态转到待处理状态，则需要清除寄存器2中的状态值。

这样，寄存器1中的状态值是1，寄存器2中的状态值是0，也就回到了待处理状态的状态值10。

本申请中，在中断控制器中，通过两个寄存器维持一个状态机，可以使中断控制器持续获取中断处理的状态，有利于在处理完成一个中断后，快速进入下一个中断的处理。

如图6所示，该第二处理器中包括硬件辅助切换装置和第二重定向装置。该硬件辅助切换装置的结构可以参阅图8进行理解。如图8所示，该硬件辅助切换装置在处理用户态中断时，可支持不进内核实现线程切换。该硬件辅助切换装置包括处理器状态判断模块，线程切换模块和内核通知模块。

处理器状态判断模块用于判断处理器当前状态是否支持立即响应用户态的中断。根据处理器的当前状态，确定通过线程切换方式或内核通知的方式，运行用户态的影子线程响应中断。

线程切换模块包括影子线程的信息，第一线程的信息，以及与线程相关的缓存数据刷新模块。其中，影子线程是运行在用户态的host中的线程，用于执行中断处理函数。第一线程是第二处理器在切换线程的信息前在运行的线程，影子线程的信息可通过其他处理器上运行的软件或硬件逻辑更新。线程信息切换时，第一线程的信息保存在第一线程信息模块中，当用户态中断处理完成时，将从该第一线程信息模块中恢复第一线程的相关信息。若缓存数据刷新模块不支持识别进程或虚拟机信息，则线程信息切换完成后，须通过缓存数据刷新模块刷新与线程相关的缓存信息；若缓存数据刷新模块支持识别进程或虚拟机信息，可不触发缓存数据刷新模块执行。

内核通知模块是在第二处理器的当前状态无法及时响应用户态中断时，此类情况包括但不限于第二处理器处理内核态中断或异常时，以及处理器处于空闲状态等，通过包括但不限于产生现有的或新增的中断或异常的方式，实现通知内核唤醒影子线程执行。

在第二处理器确定第一线程处于虚拟化特权级时，第二重定向装置完成影子线程相关状态控制信息更新，然后将虚拟化特权级切换为host的用户态。

上述处理用户态的中断请求，可以是第二处理器运行用户态的影子线程响应并处理用户态中断，如图9所示，该影子线程包括原始线程和实现双层切换机制的协程。

原始线程在被创建之后阻塞睡眠，也可被内核唤醒执行用户态中断服务例程的功能函数。

双层切换机制由硬件辅助切换装置，以及第一线程和影子线程上下文切换模块构成。若处理器可响应用户态中断时，协程运行所需信息（包括但不限于内存地址空间，数据结构等）通过硬件辅助切换装置完成切换，并通过第一线程和影子线程上下文切换模块完成第一线程和影子线程的上下文切换。影子线程的协程进一步调用用户态中断服务例程的功能函数完成用户态中断的响应与处理。

以上，从硬件功能的角度介绍了异步处理的过程，下面结合图10介绍一种中断处理的方法的过程：

301、第一处理器获取第一VM执行与虚拟I/O设备相关操作而触发的第一中断请求。

该步骤301的过程可以是第一VM在执行超出第一VM权限的操作时发生了中断，第一处理器可以通过硬件逻辑检测到该中断代码，进而获取到该第一中断请求。

该第一中断请求包括VEI ID，可以是第一VM将VEI ID写入到用于存储VEI ID的寄存器中，第一处理器从该用于存储VEI ID的寄存器中读取该VEI ID。

302、第一处理器获取第一VM的VM ID。

第一处理器可以从用于存储虚拟机标识的寄存器中获取该VM ID，该VM ID在第一处理器运行第一VM时写入到该用于存放虚拟机标识的寄存器中。

303、第一处理器向中断控制器发送VEI ID和第一VM的VM ID。

304、中断控制器根据VEI ID和第一VM的VM ID查询对应的处理器标识和VEI物理号，以及确定该第一中断请求所对应的host特权级。

该查询过程可以参阅表1进行理解。

表1：映射表

VEI ID	VM ID	CPU ID	VEI物理号	Host的特权级
VEI 1	VM 1	CPU 2	VEI 100	用户态
VEI 2	VM 1	CPU 2	VEI 101	内核态
…	…	…	…	…

由表1可知，通过VEI ID和VM ID，通过表1的映射关系，可以查询到对应的CPU ID、VEI物理号以及host特权级。

如果查找到对应的处理器标识和VEI物理号，则执行步骤305。否则，报告异常或忽略该中断请求。

305、若第一中断请求需要在host的用户态处理，则设置影子线程的信息。

影子线程的信息包括该影子线程所占用的内存地址空间，线程数据结构的信息，中断处理函数的入口地址等，该影子线程也可以称为影子线程。

306、若第一中断请求需要在host的内核态处理，则中断控制器通知第二处理器在内核态处理该第一中断请求。

该第二中断请求为用户态虚拟事件中断。

307、中断控制器向第二处理器发送第二中断请求，对应地，第二处理器接收中断控制器发送的第二中断请求。

该第二中断请求包括指示host的特权级的信息。

308、第二处理器确定第二处理器的当前状态是否支持相应用户态中断，若支持，则执行步骤309，若不支持，则执行步骤310。

309、第二处理器当前支持用户态中断，则第二处理器将第一线程的信息切换为影子线程的信息。

第一线程指的是第二处理器当前正在运行的线程，影子线程的信息是步骤305中断控制器设置的信息。

310、若不支持，则第二处理器触发异常或中断通知内核处理该虚拟事件中断。

第二处理器在内核态识别出异常或中断触发原因，然后调度唤醒虚拟I/O处理线程，虚拟I/O处理线程处理虚拟事件中断。虚拟I/O处理线程完成虚拟事件中断处理后，阻塞等待下次虚拟事件处理的通知。

若第二处理器当前处于空闲状态，则无法响应用户态中断。

311、若支持用户态中断，则第二处理器确定第一线程是否处于虚拟化特权级，若是，则执行步骤312，若否，则执行步骤315。

312、第二处理器从虚拟化特权级切换为host的用户态。

该步骤311可以是上述图6中的第二重定向装置执行的。

313、第二处理器跳转到影子线程的中断处理函数的入口地址执行与VEI物理号对应的中断处理函数。

在计算机设备中可以通过一块内存来维护VEI物理号与中断处理函数的映射关系，第二处理器可以根据该VEI物理号从该内存中读取对应的中断处理函数。

第一线程处于虚拟化特权级，则保存在第二处理器上运行的第二VM的上下文，恢复影子线程上下文。

第二处理器可以运行该用户态的线程通过系统调用陷入内核态，然后调用物理I/O设备的驱动，从而操作物理I/O设备。

314、第二处理器执行完第二中断请求后，恢复第二VM的上下文，切换回中断前的虚拟化特权级，并跳转到第二VM被打断的指令地址运行第二VM继续执行该被打断的指令。

315、若第一线程未处于虚拟化特权级，则响应用户态中断，并跳转到影子线程的中断处理函数的入口地址执行与VEI物理号对应的中断处理函数。

若未处于虚拟化特权级，则图6中的硬件辅助切换装置直接响应用户态中断，并跳转到影子线程的中断处理函数的入口地址执行第二中断请求。

第一线程没有处于虚拟化特权级，则保存第一线程上下文，恢复影子线程上下文。

316、若被中断线的程没有处于虚拟化特权级，则恢复第一线程上下文，跳转到第一线程继续执行被中断的指令。

上述异步处理方式1中使用了VEI ID、VM ID，以及VEI物理号，若VEI ID是全局唯一的，则可以简化上述图10的中断处理过程，只需要使用VEI ID确定对应的第二处理器的标识以及对应的host的特权级即可实现与上述图10对应的实施例所描述的相应过程。

本申请实施例提供的计算机系统，通过异步处理的方式将第一VM运行期间所发生的异常或中断通过中断控制器和第二处理器传递到host的用户态进行处理，避免了host的内核与用户态之间的切换开销，提高了计算机系统的性能。

异步处理方式2：发送端的流程不经过host内核，接收端的流程经过host内核。

该异步处理方2中，发送端的流程与上述异步处理方式1中的过程相同，都是采用在第一处理器和中断控制器中设置虚拟事件中断组件来实现。

不同的是：中断控制器会向第二处理器发送内核的第二中断请求。

如图11所示，第二处理器接收内核的第二中断请求后，并跳转到内核的中断处理函数的入口地址。

在跳转到host的内核态的中断处理函数的入口地址之前，若第二处理器上运行着第二VM，则先从第二VM的虚拟化特权级切换到host的内核态，若第二处理器上没运行虚拟机，host处于用户态，则先从host的用户态切换到host的内核态，若host处于内核态，则直接跳转到中断处理函数的入口地址。

内核处理该中断请求，并通过调度器唤醒虚拟I/O处理线程中的内核态线程或用户态线程，处理该第二中断请求。

若调度器唤醒的是用户态线程，则该处理过程可以是第二处理器运行该用户态的线程通过系统调用陷入内核态，然后调用物理I/O设备的驱动，从而操作物理I/O设备。

若调度器唤醒的是内核线程，则该处理过程可以运行该内核线程调用物理I/O设备的驱动，从而操作物理I/O设备。

本申请实施例相比于上述异步处理方式1中的异步虚拟IO处理框架，可省去硬件辅助切换逻辑以及影子线程技术，简化异步虚拟IO处理框架的设计和实现。

异步处理方式3：发送端的流程采用同步方式，接收端的流程不经过host内核。

该异步处理方3中，发送端流程基于同步虚拟IO处理框架中的第一重定向装置和影子执行环境，实现异步虚拟IO请求通知框架。

在该异步处理方式3中，如图12所示，中断控制器中包括用户态映射地址空间。第一VM可以通过前述同步方式中描述的方案，产生用户态的异常指令，进入用户态影子执行环境，并通过访问中断控制器中用户态映射地址空间向第二处理器发送中断请求。该中断请求可以为用户态中断或内核态中断。然后中断控制器控制该中断请求执行异步处理方1中的接收端的流程，这部分内容可以参阅异步处理方1中接收端部分的相应内容进行理解，此处不再重复赘述。

本申请实施例将发中断的流程开销缩减为上下文切换的流程开销，省去无关的虚拟机陷出流程开销，省去了异步处理方1中的虚拟事件中断组件，降低硬件设计与实现复杂度。

异步处理方式4：发送端的流程经过host内核，接收端的流程不经过host内核。

该异步处理方式4中，如图13所示，发送端流程中可以不涉及到虚拟机，中断控制器中可以包括内核映射地址空间和用户态映射地址空间。第一处理器可以运行内核态线程访问中断控制器中的内核映射地址空间触发第一中断请求，第一处理器可以运行用户态线程访问中断控制器中的用户态映射地址空间触发第一中断请求。也就是第一处理器可以通过用户态线程或内核态线程向所述中断控制器发送第一中断请求。中断控制器：设置所述影子线程的信息，向所述第二处理器发送第二中断请求。所述第二处理器：从第一线程切换至影子线程，该第一线程为第二处理器切换前运行的线程，影子线程是运行在用户态的host中的用于包含中断处理函数的线程，中断处理函数与第二中断请求对应；第二处理器具体用于，在host的用户态下执行中断处理函数。该第二处理器部分的处理过程可以参阅前述异步处理方式1中的接收端的流程进行理解，此处不再重复赘述。

本申请实施例提供的方案，通过异步处理的方式将用户态线程或内核态线程触发的中断通过中断控制器和第二处理器传递到host的用户态进行处理，避免了host的内核与用户态之间的切换开销，提高了计算机系统的性能。

在本申请的另一实施例中，还提供一种计算机可读存储介质，计算机可读存储介质中存储有计算机执行指令，当设备的处理器执行该计算机执行指令时，设备执行上述图2至图5中异常处理的方法。

在本申请的另一实施例中，还提供一种计算机可读存储介质，计算机可读存储介质中存储有计算机执行指令，当设备的处理器执行该计算机执行指令时，设备执行上述图6至图13中中断处理的方法。

在本申请的另一实施例中，还提供一种计算机程序产品，该计算机程序产品包括计算机执行指令，该计算机执行指令存储在计算机可读存储介质中；当设备的处理器执行该计算机执行指令时，设备执行上述图2至图5中异常处理的方法。

在本申请的另一实施例中，还提供一种计算机程序产品，该计算机程序产品包括计算机执行指令，该计算机执行指令存储在计算机可读存储介质中；当设备的处理器执行该计算机执行指令时，设备执行上述图6至图13中中断处理的方法。

在本申请的另一实施例中，还提供一种芯片系统，该芯片系统包括处理器，该处理器用于实现上述图2至图5中异常处理的方法。在一种可能的设计中，芯片系统还可以包括存储器，存储器，用于保存异常处理的必要的程序指令和数据。该芯片系统，可以由芯片构成，也可以包含芯片和其他分立器件。

在本申请的另一实施例中，还提供一种芯片系统，该芯片系统包括处理器，该处理器用于执行上述图6至图13中中断处理的方法。在一种可能的设计中，芯片系统还可以包括存储器，存储器，用于保存中断处理的程序指令和数据。该芯片系统，可以由芯片构成，也可以包含芯片和其他分立器件。

本领域普通技术人员可以意识到，结合本文中所公开的实施例描述的各示例的单元及算法步骤，能够以电子硬件、或者计算机软件和电子硬件的结合来实现。这些功能究竟以硬件还是软件方式来执行，取决于技术方案的特定应用和设计约束条件。专业技术人员可以对每个特定的应用来使用不同方法来实现所描述的功能，但是这种实现不应认为超出本申请实施例的范围。

所属领域的技术人员可以清楚地了解到，为描述的方便和简洁，上述描述的系统、装置和单元的具体工作过程，可以参考前述方法实施例中的对应过程，在此不再赘述。

在本申请实施例所提供的几个实施例中，应该理解到，所揭露的系统、装置和方法，可以通过其它的方式实现。例如，以上所描述的装置实施例仅仅是示意性的，例如，单元的划分，仅仅为一种逻辑功能划分，实际实现时可以有另外的划分方式，例如多个单元或组件可以结合或者可以集成到另一个系统，或一些特征可以忽略，或不执行。另一点，所显示或讨论的相互之间的耦合或直接耦合或通信连接可以是通过一些接口，装置或单元的间接耦合或通信连接，可以是电性，机械或其它的形式。

作为分离部件说明的单元可以是或者也可以不是物理上分开的，作为单元显示的部件可以是或者也可以不是物理单元，即可以位于一个地方，或者也可以分布到多个网络单元上。可以根据实际的需要选择其中的部分或者全部单元来实现本实施例方案的目的。

另外，在本申请实施例各个实施例中的各功能单元可以集成在一个处理单元中，也可

以是各个单元单独物理存在，也可以两个或两个以上单元集成在一个单元中。

功能如果以软件功能单元的形式实现并作为独立的产品销售或使用时，可以存储在一个计算机可读取存储介质中。基于这样的理解，本申请实施例的技术方案本质上或者说对现有技术做出贡献的部分或者该技术方案的部分可以以软件产品的形式体现出来，该计算机软件产品存储在一个存储介质中，包括若干指令用以使得一台计算机设备（可以是个人计算机，服务器，或者网络设备等）执行本申请实施例各个实施例方法的全部或部分步骤。而前述的存储介质包括：U 盘、移动硬盘、只读存储器（Read-Only Memory，ROM）、随机存取存储器（Random Access Memory，RAM）、磁碟或者光盘等各种可以存储程序代码的介质。

权 利 要 求

1.一种计算机设备，其特征在于，所述计算机设备包括硬件层，所述硬件层上运行有宿主机host，所述硬件层包括第一处理器，所述第一处理器上运行有虚拟机VM，所述第一处理器包括重定向装置；

所述第一处理器用于：获取所述VM触发的异常指令；

所述重定向装置被配置为：

从所述VM的虚拟化特权级切换到所述host的用户态；

所述第一处理器还用于：在host的用户态下执行所述异常指令对应的异常处理函数。

2.根据权利要求1所述的设备，其特征在于，

所述重定向装置还被配置为：将所述异常处理函数的入口地址写入到第一寄存器中；

所述第一处理器，具体用于从所述第一寄存器中读取所述异常处理函数的入口地址，并根据所述入口地址调用所述异常处理函数，在所述host的用户态下执行所述异常指令对应的异常处理函数。

3.根据权利要求1或2所述的计算机设备，其特征在于，

所述异常指令是所述VM执行输入/输出I/O操作触发的。

4.根据权利要求1-3任一项所述的计算机设备，其特征在于，所述异常处理函数包含于运行在host的用户态的虚拟处理器的线程中，所述虚拟处理器的线程被分配的执行环境包括主线程执行环境、所述VM的执行环境和影子执行环境，所述影子执行环境具有独立于所述主线程执行环境和所述VM的执行环境的用户态内存空间；所述影子执行环境用于执行所述异常处理函数，所述异常处理函数用于处理触发所述异常指令的异常；

所述第一处理器具体用于：从所述VM的上下文切换为所述用户态的所述影子执行环境的上下文，所述影子执行环境的用户态内存空间用于存储在切换上下文时移出的所述VM的上下文。

5.一种计算机设备，其特征在于，所述计算机设备包括硬件层，所述硬件层上运行有宿主机host，所述硬件层包括第一处理器、第二处理器和中断控制器，所述第一处理器上运行有虚拟机第一VM，所述第一处理器包括虚拟事件通知装置；

所述第一处理器用于：获取所述第一VM触发的第一中断请求，所述第一中断请求用于请求执行中断处理函数；

所述虚拟事件通知装置被配置为：响应于所述第一中断请求，向所述中断控制器发送中断信息；

所述中断控制器被配置为：根据所述中断信息确定所述第二处理器的标识，根据所述第二处理器的标识向所述第二处理器发送第二中断请求，所述第二中断请求用于请求所述第二处理器执行所述中断处理函数；

所述第二处理器用于：响应于接收到所述第二中断请求，执行所述中断处理函数。

6.根据权利要求5所述的计算机设备，其特征在于，所述第一中断请求包括虚拟事件中断请求号VEI ID，所述VEI ID用于标识触发所述第一中断请求的虚拟事件；

所述第一处理器还用于：获取所述第一VM的VM ID；

所述中断信息包括所述VEI ID和所述第一VM的VM ID；

所述中断控制器具体被配置为：根据所述VEI ID和所述第一VM的VM ID确定对应的所述第二处理器的标识、VEI物理号，以及处理所述第一中断请求的所述host的特权级，所述VEI物理号用于在所述host中标识所述虚拟事件；根据所述第二处理器的标识向所述第二处理器发送第二中断请求，所述第二中断请求包括指示所述host的特权级的信息；

所述第二处理器具体用于：获取所述VEI物理号，根据所述host的特权级对应的处理方式，响应于接收到所述第二中断请求，执行与所述VEI物理号对应的中断处理函数。

7.根据权利要求5所述的计算机设备，其特征在于，所述第一中断请求包括虚拟事件中断请求号VEI ID，所述VEI ID用于标识触发所述第一中断请求的虚拟事件；

所述中断信息包括所述VEI ID；

所述中断控制器具体被配置为：根据所述VEI ID确定对应的所述第二处理器的标识，以及处理所述第一中断请求的所述host的特权级；根据所述第二处理器的标识向所述第二处理器发送第二中断请求，所述第二中断请求包括指示所述host的特权级的信息；

所述第二处理器具体用于：获取所述VEI ID，根据所述host的特权级对应的处理方式，响应于接收到所述第二中断请求，执行与所述VEI ID对应的中断处理函数。

8.根据权利要求6或7所述的计算机设备，其特征在于，所述host的特权级为用户态；所述第二处理器还包括硬件辅助切换装置；

所述硬件辅助切换装置被配置为：

从第一线程切换至影子线程，所述第一线程为所述第二处理器切换前运行的线程，所述影子线程是运行在用户态的所述host中的线程，所述影子线程用于执行所述中断处理函数；

所述第二处理器具体用于，在所述host的用户态下执行所述中断处理函数。

9.根据权利要求8所述的计算机设备，其特征在于，所述第二处理器还包括重定向装置；

所述第二处理器，还用于在所述host的用户态下执行所述中断处理函数之前，确定所述第一线程是否处于第二VM的虚拟化特权级，所述第二VM运行在所述第二处理器上；

所述重定向装置被配置为：若所述被中断的线程处于第二VM的虚拟化特权级，则将所述第二VM的虚拟化特权级切换为所述host的用户态。

10.根据权利要求9所述的计算机设备，其特征在于，

所述重定向装置还被配置为：将所述中断处理函数的入口地址写入到第一寄存器中；

所述第二处理器，具体用于从所述第一寄存器中读取所述中断处理函数的入口地址，并根据所述入口地址调用所述中断处理函数，在所述host的用户态下执行所述中断处理函数。

11.根据权利要求9或10所述的计算机设备，其特征在于，

所述第二处理器具体用于：

从所述第二VM的上下文切换为所述影子线程的上下文，并在所述host的用户态下处理中断；

在处理完中断后，从所述影子线程的上下文切换回所述第二VM的上下文；

所述重定向装置还被配置为：从所述host的用户态切换回所述第二VM的虚拟化特权级；

所述第二处理器还用于：继续运行所述第二VM。

12.根据权利要求8所述的计算机设备，其特征在于，

所述第二处理器还用于：

从所述第一线程的上下文切换为所述影子线程的上下文,并在所述host的用户态下处理中断；

在处理完中断后，从所述影子线程的上下文切换为所述第一线程的上下文；

继续执行所述第一线程。

13.根据权利要求6或7所述的计算机设备，其特征在于，所述host的特权级为内核态，

所述第二处理器用于：

接收所述中断控制器发送的所述host的内核态的所述第二中断请求；

跳转到所述host的内核态的中断处理函数的入口地址，所述中断处理函数与所述第二中断请求对应；

在所述host的内核态下执行所述内核态的所述中断处理函数。

14.一种异常处理的方法，其特征在于，所述方法应用于计算机设备，所述计算机设备上运行有宿主机host和虚拟机VM，所述方法包括：

获取所述VM触发的异常指令；

从所述VM的虚拟化特权级切换到所述host的用户态；

在host的用户态下执行所述异常指令对应的异常处理函数。

15.根据权利要求14所述的方法，其特征在于，所述方法还包括：

将所述异常处理函数的入口地址写入到第一寄存器中；

所述在host的用户态下执行所述异常指令对应的异常处理函数，包括：

从所述第一寄存器中读取所述异常处理函数的入口地址，并根据所述入口地址调用所述异常处理函数，在所述host的用户态下执行所述异常指令对应的异常处理函数。

16.根据权利要求14或15所述的方法，其特征在于，所述异常指令是所述VM执行输入/输出I/O操作触发的。

17.根据权利要求14-16任一项所述的方法，其特征在于，所述在host的用户态下执行所述异常指令对应的异常处理函数，包括：

从所述VM的上下文切换为所述用户态的影子执行环境的上下文，并在所述host的用户态下处理异常；其中，所述异常处理函数包含于运行在host的用户态的虚拟处理器的线程中，所述虚拟处理器的线程被分配的执行环境包括主线程执行环境、所述VM的执行环境和影子执行环境，所述影子执行环境具有独立于所述主线程执行环境和所述VM的执行环境的用户态内存空间；所述影子执行环境用于执行所述异常处理函数，所述异常处理函数用于处理触发所述异常指令的异常。

18.一种中断处理的方法，其特征在于，所述方法应用于计算机设备，所述计算机设备上运行有宿主机host和第一VM；所述方法包括：

获取所述第一VM触发的第一中断请求，所述第一中断请求用于请求执行中断处理函数；

响应于所述第一中断请求，确定中断信息；

根据所述中断信息生成第二中断请求，所述第二中断请求用于请求执行所述中断处理函数；

响应于所述第二中断请求，执行所述中断处理函数。

19.根据权利要求18所述的方法，其特征在于，所述第一中断请求包括虚拟事件中断请求号VEI ID，所述VEI ID用于标识触发所述第一中断请求的虚拟事件；所述方法还包括：

获取所述第一VM的VM ID，所述中断信息包括所述VEI ID和所述第一VM的VM ID；

根据所述VEI ID和所述第一VM的VM ID确定对应的VEI物理号，以及处理所述第一中断请求的所述host的特权级，所述VEI物理号用于在所述host中标识所述虚拟事件；

所述响应于所述第二中断请求，执行所述中断处理函数，包括：

获取所述VEI物理号，根据所述host的特权级对应的处理方式，响应于接收到所述第二中断请求，执行与所述VEI物理号对应的中断处理函数。

20.根据权利要求18所述的方法，其特征在于，所述第一中断请求包括虚拟事件中断请求号VEI ID，所述VEI ID用于标识触发所述第一中断请求的虚拟事件，所述中断信息包括所述VEI ID；所述方法还包括：

根据所述VEI ID确定处理所述第一中断请求的所述host的特权级；

所述响应于所述第二中断请求，执行所述中断处理函数，包括：

获取所述VEI ID，根据所述host的特权级对应的处理方式，响应于接收到所述第二中断请求，执行与所述VEI ID对应的中断处理函数。

21.根据权利要求19或20所述的方法，其特征在于，所述host的特权级为用户态；所述响应于所述第二中断请求，执行所述中断处理函数，包括：

从第一线程切换至影子线程，所述第一线程为所述宿主机切换前运行的线程，所述影子线程是运行在用户态的所述host中的线程，所述影子线程用于执行所述中断处理函数；

在所述host的用户态下执行所述中断处理函数。

22.根据权利要求21所述的方法，其特征在于，所述方法还包括：

在所述host的用户态下执行所述中断处理函数之前，确定所述第一线程是否处于第二VM的虚拟化特权级，所述第二VM运行在所述计算机设备上；

若所述被中断的线程处于第二VM的虚拟化特权级，则将所述第二VM的虚拟化特权级切换为所述host的用户态。

23.根据权利要求22所述的方法，其特征在于，所述方法还包括：

将所述中断处理函数的入口地址写入到第一寄存器中；

从所述第一寄存器中读取所述中断处理函数的入口地址，并根据所述入口地址调用所述中断处理函数，在所述host的用户态下执行所述中断处理函数。

24.根据权利要求22或23所述的方法，其特征在于，所述在所述host的用户态下执行所述中断处理函数，包括：

从所述第二VM的上下文切换为所述影子线程的上下文,并在所述host的用户态下处理中断；

在处理完中断后，从所述影子线程的上下文切换回所述第二VM的上下文；

从所述host的用户态切换回所述第二VM的虚拟化特权级；

继续运行所述第二VM。

25.根据权利要求21所述的方法，其特征在于，所述方法还包括：

从所述第一线程的上下文切换为所述影子线程的上下文,并在所述host的用户态下处理中断；

在处理完中断后，从所述影子线程的上下文切换为所述第一线程的上下文；

继续执行所述第一线程。

26.根据权利要求19或20所述的方法，其特征在于，所述host的特权级为内核态，所述根据所述host的特权级对应的处理方式，处理第二中断请求，以执行与所述VEI物理号对应的操作，包括：

跳转到所述host的内核态的中断处理函数的入口地址，所述中断处理函数与所述第二中断请求对应；

在所述host的内核态下执行所述内核态的所述中断处理函数。

27.一种计算机可读存储介质，其上存储有计算机程序，其特征在于，所述计算机程序被处理器执行时实现如权利要求14-17任一项所述的方法。

28.一种计算机可读存储介质，其上存储有计算机程序，其特征在于，所述计算机程序被处理器执行时实现如权利要求18-26任一项所述的方法。

29.一种芯片系统，其特征在于，包括处理器，所述处理器被调用用于执行如权利要求14-17任一项所述的方法。

30.一种芯片系统，其特征在于，包括处理器，所述处理器被调用用于执行如权利要求18-26任一项所述的方法。

31.一种计算机程序产品，其特征在于，包括计算机程序，所述计算机程序当被一个或多个处理器执行时用于实现如权利要求14-17任一项所述的方法。

32.一种计算机程序产品，其特征在于，包括计算机程序，所述计算机程序当被一个或多个处理器执行时用于实现如权利要求 18-26 任一项所述的方法。