(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 113127263 B
(45)授权公告日 2023.04.07

(21)申请号 202010042191.3
(22)申请日 2020.01.15
(65)同一申请的已公布的文献号
申请公布号 CN 113127263 A
(43)申请公布日 2021.07.16
(73)专利权人 中移(苏州)软件技术有限公司
地址 215163 江苏省苏州市高新区昆仑山路58号1幢
专利权人 中国移动通信集团有限公司
(72)发明人 丁翔
(74)专利代理机构 北京派特恩知识产权代理有限公司 11270
专利代理师 姚文娴 张颖玲
(51)Int.Cl.
G06F 11/14(2006.01)

审查员 刘升

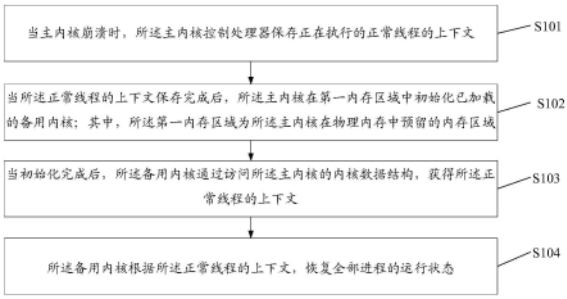
权利要求书3页 说明书14页 附图7页

(54)发明名称

一种内核崩溃恢复方法、装置、设备及存储介质

(57)摘要

本申请实施例公开了一种内核崩溃恢复方法、装置、设备及存储介质，其中，所述方法包括：当主内核崩溃时，所述主内核控制处理器保存正在执行的正常线程的上下文；当所述正常线程的上下文保存完成后，所述主内核在第一内存区域中初始化已加载的备用内核；其中，所述第一内存区域为所述主内核在物理内存中预留的内存区域；当初始化完成后，所述备用内核通过访问所述主内核的内核数据结构，获得所述正常线程的上下文；所述备用内核根据所述正常线程的上下文，恢复全部进程的运行状态。本申请实施例中，通过主备内核切换的方式，在主内核崩溃后直接启动备用内核，不需要通过执行BIOS程序来重启整个操作系统，可以节省系统恢复时间，并避免数据丢失。



CN 113127263 B

1.一种内核崩溃恢复方法,其特征在于,所述方法包括:

当主内核崩溃时,所述主内核控制处理器保存正在执行的正常线程的上下文;

当所述正常线程的上下文保存完成后,所述主内核在第一内存区域中初始化已加载的备用内核;其中,所述第一内存区域为所述主内核在物理内存中预留的内存区域;

当初始化完成后,所述备用内核通过访问所述主内核的内核数据结构,获得所述正常线程的上下文;

所述备用内核根据所述正常线程的上下文,恢复全部进程的运行状态;

其中,所述正常线程的上下文包括所述主内核的进程描述符链表、内存区域描述符表、虚拟内存页、文件描述符表;

对应地,所述备用内核根据所述正常线程的上下文,恢复全部进程的运行状态,包括:

所述备用内核访问所述主内核的进程描述符链表,读取待恢复的进程列表;

所述备用内核为所述待恢复的进程列表中的每一待恢复进程,创建一个新进程;

所述备用内核根据所述主内核的内存区域描述符列表的内容,在每一所述新进程中分别恢复对应的待恢复进程的用户内存空间;

所述备用内核根据所述主内核的每个虚拟内存页的内容,在每一所述新进程中分别恢复对应的待恢复进程的硬件页表和交换页表;

所述备用内核根据所述主内核的文件描述符表的内容,在每一所述新进程中恢复打开的文件。

2.根据权利要求1所述的方法,其特征在于,所述主内核控制处理器保存正在执行的正常线程的上下文之前,所述方法还包括:

处理器上电后,引导程序基于主内核文件,加载并初始化主内核;

所述主内核基于备用内核文件,在所述第一内存区域中加载备用内核;其中,所述主内核文件与所述备用内核文件中的内存偏移量不同。

3.根据权利要求2所述的方法,其特征在于,所述方法还包括:

当所述全部进程的运行状态恢复后,所述备用内核作为新的主内核运行,回收可用的物理内存;

所述新的主内核在物理内存中预留第二内存区域,并轮流基于所述主内核文件和所述备用内核文件二者之一,在所述第二内存区域中加载新的备用内核;其中,所述新的主内核对应的内核文件与所述新的备用内核对应的内核文件不同。

4.根据权利要求1所述的方法,其特征在于,

所述处理器包括正常处理器和由主内核崩溃导致崩溃的处理器;

对应地,所述当主内核崩溃时,所述主内核控制处理器保存正在执行的正常线程的上下文,包括:

当主内核崩溃时,所述主内核通过错误处理机制,控制所述崩溃的处理器向正常处理器发送不可屏蔽中断;

所述主内核控制每一所述正常处理器接收所述不可屏蔽中断后,保存正在执行的线程的上下文;

所述当所述正常线程的上下文保存完成后,所述主内核在第一内存区域中初始化已加载的备用内核,包括:

所述主内核控制每一所述正常处理器在保存完正在执行的线程的上下文后停止运行；

所述主内核检测到全部正常处理器停止后，控制所述崩溃的处理器执行所述备用内核的初始化程序，在第一内存区域中初始化已加载的备用内核。

5.根据权利要求1所述的方法，其特征在于，所述备用内核根据所述主内核的内存区域描述符列表的内容，在每一所述新进程中分别恢复对应的待恢复进程的用户内存空间，包括：

所述备用内核访问所述主内核中每一所述待恢复进程的内存区域描述符列表；

所述备用内核根据每一所述待恢复进程的内存区域描述符列表中的每一内存区域描述符，分别为对应的所述新进程创建一个具有相同属性的新内存描述符。

6.根据权利要求5所述的方法，其特征在于，所述备用内核根据所述主内核的内存区域描述符列表的内容，在每一所述新进程中分别恢复对应的待恢复进程的用户内存空间，还包括：

当所述内存区域描述符中存在内存映射文件时，所述备用内核在与所述内存区域描述符对应的新进程中打开所述文件；

所述备用内核将所述文件重新映射到所述备用内核中相应的内存区域。

7.根据权利要求6所述的方法，其特征在于，

所述正常线程的上下文还包括所述主内核的交换区域描述符；

对应地，所述备用内核在与所述内存区域描述符对应的新进程中打开所述文件，包括：

当所述文件映射的内存区域在交换分区时，所述备用内核检索所述主内核的交换区域描述符，获得指向与所述文件对应的文件结构的指针；

所述备用内核通过所述文件结构的指针，获得所述文件结构的内容；

所述备用内核根据所述文件结构的内容，将所述文件重新打开，并将所述文件重新映射到所述备用内核中相应的内存区域。

8.根据权利要求1所述的方法，其特征在于，

所述虚拟内存页包括每一所述待恢复进程的硬件页表和交换页表；

对应地，所述备用内核根据所述主内核的每个虚拟内存页的内容，在每一所述新进程中分别恢复对应的待恢复进程的硬件页表和交换页表，包括：

当所述备用内核从所述待恢复进程的硬件页表中检索到存在条目时，所述备用内核为每一所述条目在所述备用内核中分配一个新页，并将每一所述条目相应页的内容分别复制到对应的新页；

所述备用内核为所述主内核交换到磁盘的每个页面，在所述备用内核的交换分区中分配一个新页，并将所述主内核交换到磁盘的每个页面的内容分别复制到对应的新页。

9.根据权利要求1所述的方法，其特征在于，所述备用内核根据所述主内核的文件描述符表的内容，在每一所述新进程中恢复打开的文件，包括：

所述备用内核访问所述主内核的文件描述符表，读取每一所述待恢复的进程中文件的名称、位置、打开标志和当前偏移量；

所述备用内核根据每一所述待恢复的进程中文件的名称、位置、打开标志和当前偏移量，分别在对应的所述新进程中恢复所述文件的打开状态。

10.一种内核崩溃恢复装置，其特征在于，所述装置包括：

主内核，用于：

当主内核崩溃时，控制处理器保存正在执行的正常线程的上下文；

当所述正常线程的上下文保存完成后，在第一内存区域中初始化已加载的备用内核；其中，所述第一内存区域为所述主内核在物理内存中预留的内存区域；

备用内核，用于：

当初始化完成后，通过访问所述主内核的内核数据结构，获得所述正常线程的上下文；

根据所述正常线程的上下文，恢复全部进程的运行状态；

其中，所述备用内核，还用于：

访问所述主内核的进程描述符链表，读取待恢复的进程列表；

所述待恢复的进程列表中的每一待恢复进程，创建一个新进程；

为所述待恢复的进程列表中的每一待恢复进程，创建一个新进程；

根据所述主内核的内存区域描述符列表的内容，在每一所述新进程中分别恢复对应的待恢复进程的用户内存空间；

根据所述主内核的每个虚拟内存页的内容，在每一所述新进程中分别恢复对应的待恢复进程的硬件页表和交换页表；

根据所述主内核的文件描述符表的内容，在每一所述新进程中恢复打开的文件。

11.一种计算机设备，包括存储器和处理器，所述存储器存储有可在处理器上运行的计算机程序，其特征在于，所述处理器执行所述程序时实现权利要求1至9任一项所述方法中的步骤。

12.一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，该计算机程序被处理器执行时实现权利要求1至9任一项所述方法中的步骤。

一种内核崩溃恢复方法、装置、设备及存储介质

技术领域

[0001] 本申请实施例涉及但不限于计算机领域，尤其涉及一种内核崩溃恢复方法、装置、设备及存储介质。

背景技术

[0002] 现代的操作系统越来越复杂，就不可避免的会出现各种错误，而操作系统的内核错误是比较严重的一种。一旦内核出现严重错误，比如内核崩溃时，操作系统默认的操作都是重启整个系统，而重启系统会导致：1)业务中断；2)可能会丢失数据。

发明内容

[0003] 有鉴于此，本申请实施例提供一种内核崩溃恢复方法、装置、设备及存储介质。

[0004] 本申请实施例的技术方案是这样实现的：

[0005] 一方面，本申请实施例提供一种内核崩溃恢复方法，所述方法包括：

[0006] 当主内核崩溃时，所述主内核控制处理器保存正在执行的正常线程的上下文；

[0007] 当所述正常线程的上下文保存完成后，所述主内核在第一内存区域中初始化已加载的备用内核；其中，所述第一内存区域为所述主内核在物理内存中预留的内存区域；

[0008] 当初始化完成后，所述备用内核通过访问所述主内核的内核数据结构，获得所述正常线程的上下文；

[0009] 所述备用内核根据所述正常线程的上下文，恢复全部进程的运行状态。

[0010] 另一方面，本申请实施例提供一种内核崩溃恢复装置，所述装置包括：

[0011] 主内核，用于：

[0012] 当主内核崩溃时，控制处理器保存正在执行的正常线程的上下文；

[0013] 当所述正常线程的上下文保存完成后，在第一内存区域中初始化已加载的备用内核；其中，所述第一内存区域为所述主内核在物理内存中预留的内存区域；

[0014] 备用内核，用于：

[0015] 当初始化完成后，通过访问所述主内核的内核数据结构，获得所述正常线程的上下文；

[0016] 根据所述正常线程的上下文，恢复全部进程的运行状态。

[0017] 再一方面，本申请实施例提供一种计算机设备，包括存储器和处理器，所述存储器存储有可在处理器上运行的计算机程序，所述处理器执行所述程序时实现上述方法中的步骤。

[0018] 再一方面，本申请实施例提供一种计算机可读存储介质，其上存储有计算机程序，该计算机程序被处理器执行时实现上述方法中的步骤。

[0019] 本申请实施例中，通过主备内核切换的方式，在主内核崩溃后直接启动备用内核，不需要通过执行基本输入输出系统(Basic Input Output System，BIOS)程序来重启整个操作系统，可以节省系统恢复时间。此外，由于不执行BIOS程序，备用内核初始化后，主内核

的内核数据结构还保存在内存中，备用内核通过访问主内核的内核数据结构，可以获得主内核崩溃时保存的正常线程的上下文，从而恢复并继续运行崩溃时运行的全部进程，进而避免业务中断以及数据的丢失。

附图说明

[0020] 图1A为本申请实施例提供的一种内核崩溃恢复方法的实现流程示意图；

[0021] 图1B为正常启动后的系统运行状态示意图；

[0022] 图1C为主内核崩溃后的系统运行状态示意图；

[0023] 图1D为备用内核恢复进程时的系统运行状态示意图；

[0024] 图2为本申请实施例提供的一种内核崩溃恢复方法的实现流程示意图；

[0025] 图3A为本申请实施例提供的一种内核崩溃恢复方法的实现流程示意图；

[0026] 图3B为恢复完成后的系统运行状态示意图；

[0027] 图4为本申请实施例提供的一种内核崩溃恢复方法的实现流程示意图；

[0028] 图5为本申请实施例提供的一种内核崩溃恢复方法的实现流程示意图；

[0029] 图6为本申请实施例提供的一种内核崩溃恢复方法的实现流程示意图；

[0030] 图7为本申请实施例内核崩溃恢复装置的组成结构示意图；

[0031] 图8为本申请实施例中计算机设备的一种硬件实体示意图。

具体实施方式

[0032] 为了使本申请的目的、技术方案和优点更加清楚，下面结合附图和实施例对本申请的技术方案进一步详细阐述，所描述的实施例不应视为对本申请的限制，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其它实施例，都属于本申请保护的范围。

[0033] 在以下的描述中，涉及到“一些实施例”，其描述了所有可能实施例的子集，但是可以理解，“一些实施例”可以是所有可能实施例的相同子集或不同子集，并且可以在不冲突的情况下相互结合。

[0034] 如果申请文件中出现“第一/第二”的类似描述则增加以下的说明，在以下的描述中，所涉及的术语“第一\第二\第三”仅仅是区别类似的对象，不代表针对对象的特定排序，可以理解地，“第一\第二\第三”在允许的情况下可以互换特定的顺序或先后次序，以使这里描述的本申请实施例能够以除了在这里图示或描述的以外的顺序实施。

[0035] 除非另有定义，本文所使用的所有的技术和科学术语与属于本申请的技术领域的技术人员通常理解的含义相同。本文中所使用的术语只是为了描述本申请实施例的目的，不是旨在限制本申请。

[0036] 针对背景技术中内核崩溃时系统重启存在的问题，在相关技术中有以下两种解决方案：

[0037] 1)定期保存检查点；

[0038] 该方案做法是定时将缓存数据刷回磁盘，生成快照，并将检查点写入日志文件，用于在宕机后系统重启时进行数据恢复。

[0039] 2)基于微内核的组件隔离。

[0040] 由于微内核操作系统是采用组件的方式，该方案做法是当有组件出现错误时，重启错误组件，而不需要重启整个系统。

[0041] 在上述两种方案中，仍然存在以下缺点：定期保存检查点会额外增加系统开销，影响性能，并且当恢复数据时无法恢复最近一次保存检查点到发生错误这段时间的数据，造成数据丢失；而微内核操作系统的组件模式无法应用到非微内核架构的操作系统，比如Linux、Windows，并且错误有可能扩散到其他组件，造成更多的组件发生错误。

[0042] 本申请实施例提供一种内核崩溃恢复方法，可以解决上述相关技术中存在的系统开销增加、数据丢失、适用面窄以及错误传染的问题。图1A为本申请实施例提供的内核崩溃恢复方法的实现流程示意图，如图1A所示，该方法包括：

[0043] 步骤S101，当主内核崩溃时，所述主内核控制处理器保存正在执行的正常线程的上下文；

[0044] 这里，正常线程为除了发生崩溃的线程之外的全部线程，线程的上下文包括线程运行时传递给内核的变量、参数以及线程的寄存器值等。系统恢复时可以通过检索线程的上下文，像常规上下文切换一样继续执行该上下文对应的线程。

[0045] 在实施时，可以通过对操作系统默认的崩溃处理机制进行修改，在主内核崩溃时，不进行重新引导，而是执行自定义的崩溃处理程序，保存正在执行的正常线程的上下文，并进行后续操作。

[0046] 步骤S102，当所述正常线程的上下文保存完成后，所述主内核在第一内存区域中初始化已加载的备用内核；其中，所述第一内存区域为所述主内核在物理内存中预留的内存区域；

[0047] 这里，主内核可以在启动时预留第一内存区域，所述主内核在启动后将备用内核加载至第一内存区域，但不做初始化。只要主内核运行正常，备用内核就会保留在第一内存区域中，并且它的代码永远不会执行。图1B为正常启动后的系统运行状态示意图，如图1B所示，此时主内核111、硬件121以及进程131、132、133均处于正常运行状态，备用内核112处于未运行状态。

[0048] 主内核崩溃时，当所述正常线程的上下文保存完成后，主内核跳转到备用内核的初始化入口点，控制处理器开始执行备用内核的初始化程序。此时，系统不需要重启，备用内核开始在预留的第一内存区域中初始化，主内核不再执行任何代码。图1C为主内核崩溃后的系统运行状态示意图，如图1C所示，此时主内核111处于崩溃状态，进程131、132、133处于停止运行状态，备用内核112与硬件121处于正常运行状态。

[0049] 在一些实施例中，在加载完备用内核后，主内核还可以对第一内存区域进行内存保护，以避免加载的备用内核被其他进程修改。在实施时，可根据实际需求，对第一内存区域的保护属性进行设置，包括但不限于将保护属性设置为不可写且不可执行，和/或不可访问。相应地，主内核崩溃时，当所述正常线程的上下文保存完成后，主内核需要将第一内存区域的内存保护解除，使得主内核可以访问到备用内核的初始化入口点，且备用内核的初始化代码可执行。在实施时，主内核可以对第一内存区域的保护属性进行设置，包括但不限于将保护属性设置为可写可执行，和/或可访问。

[0050] 在实施的时候，预留的第一内存区域的大小可以根据内核加载及初始化时实际需要的内存大小设置，本实施例对此不作限定。

[0051] 此外，在备用内核初始化时，可以通过修改启动代码中的内核启动参数来动态修改所述备用内核的可用物理内存大小。在一些实施例中，备用内核的启动代码中还需要分配额外的内存页面描述符，这些描述符会在备用内核恢复过程中使用。在一些实施例中，为了不破坏主内核交换出的任何页面，可以在系统中划分两个交换分区：一个由主内核使用，另一个由备用内核使用。内核启动时使用的交换分区可以由系统启动脚本根据内核版本进行选择。在实施时，可以将内核版本区分为主内核和备用内核两种版本，主内核崩溃后，备用内核在恢复过程中会根据主内核的交换分区中的数据在备用内核对应的交换分区中重新生成一份对应的新的数据。

[0052] 步骤S103，当初始化完成后，所述备用内核通过访问所述主内核的内核数据结构，获得所述正常线程的上下文；

[0053] 步骤S104，所述备用内核根据所述正常线程的上下文，恢复全部进程的运行状态。

[0054] 这里，备用内核在完成初始化之后，将开始恢复阶段。在恢复阶段，备用内核将访问主内核的内核数据结构，获得主内核崩溃时保存的正常线程的上下文。通过检索线程的上下文，备用内核可以像常规上下文切换一样继续执行该上下文对应的线程，从而恢复对应的应用程序的进程。默认情况下，备用内核会恢复全部进程。在一些实施例中，可以在屏幕上为用户提供选择需要恢复的进程的操作页面，让用户选择需要恢复的进程。在一些实施例中，还可以通过生成配置文件写入需要恢复的进程，启动脚本可以读取配置文件来恢复对应的进程，从而实现在无人值守时恢复。图1D为备用内核恢复进程时的系统运行状态示意图，如图1D所示，此时主内核111处于崩溃状态，进程131、132处于等待恢复状态，进程133正在逐步恢复至崩溃时的运行状态，备用内核112与硬件121处于正常运行状态。

[0055] 本申请实施例提供的内核崩溃恢复方法，通过主备内核切换的方式，在主内核崩溃后直接启动备用内核，不需要通过执行BIOS程序来重启整个操作系统，可以节省系统恢复时间。此外，由于不执行BIOS程序，备用内核初始化后，主内核的内核数据结构还保存在内存中，备用内核通过访问主内核的内核数据结构，可以获得主内核崩溃时保存的正常线程的上下文，从而恢复并继续运行崩溃时运行的全部进程，进而避免业务中断以及数据的丢失。

[0056] 与相关技术相比，本申请实施例具有如下的技术优点：相关技术需要重启整个操作系统，比较耗时，本申请实施例只需要重启内核，可以节省系统恢复时间；相关技术需要额外系统资源开销，对系统性能有一定影响，本申请实施例对系统运行时的性能无影响；相关技术可能无法恢复所有数据，本申请实施例可以恢复所有数据；相关技术只适用于微内核架构的操作系统，本申请实施例对微内核架构和非微内核架构的操作系统均适用；相关技术中采用微内核的组件模式时存在错误传染的问题，本申请实施例通过重启内核，可以避免错误的传染。

[0057] 本申请实施例提供一种内核崩溃恢复方法，图2为本申请实施例提供的内核崩溃恢复方法的实现流程示意图，如图2所示，该方法包括：

[0058] 步骤S201，处理器上电后，引导程序基于主内核文件，加载并初始化主内核；

[0059] 这里，主内核文件为磁盘上存储的已经编译好的操作系统内核文件，处理器上电后，可以通过执行引导程序将磁盘上的主内核文件加载至内存，并在设定的内存区域进行主内核的初始化。

[0060] 由于本申请实施例提供的内核崩溃恢复方法是通过切换备用内核进行恢复，对于已经崩溃的主内核不再需要恢复。因此，在实施的时候，可以对操作系统标准的内核源码进行修改，减少系统恢复时必须检索以及依赖的数据结构，编译出更加精简的内核文件作为主内核文件。

[0061] 步骤S202，所述主内核基于备用内核文件，在所述第一内存区域中加载备用内核；其中，所述主内核文件与所述备用内核文件中的内存偏移量不同；

[0062] 这里，备用内核文件也为磁盘上存储的已经编译好的操作系统内核文件。备用内核文件与主内核文件相比，只有内存偏移量是不同的，也就是说，基于二者加载并初始化的内核，仅内核线性空间的起始位置不同。

[0063] 在实施的时候，所述主内核加载备用内核的方法，可由本领域技术人员根据实际场景自由选择，本申请实施例对此并不做限定。例如，当所述主内核文件与所述备用内核文件为Linux内核文件时，所述主内核可以基于备用内核文件，利用kdump机制加载备用内核。

[0064] 步骤S203，当主内核崩溃时，所述主内核控制处理器保存正在执行的正常线程的上下文；

[0065] 步骤S204，当所述正常线程的上下文保存完成后，所述主内核在第一内存区域中初始化已加载的备用内核；其中，所述第一内存区域为所述主内核在物理内存中预留的内存区域；

[0066] 步骤S205，当初始化完成后，所述备用内核通过访问所述主内核的内核数据结构，获得所述正常线程的上下文；

[0067] 步骤S206，所述备用内核根据所述正常线程的上下文，恢复全部进程的运行状态。

[0068] 这里，需要说明的是，步骤S203至S206可以采用与前述步骤S101至S104同样的方式实施，在此不再赘述。

[0069] 本申请实施例提供的内核崩溃恢复方法，通过切换备用内核进行系统恢复，对于已经崩溃的主内核不再需要恢复，这样，主内核文件及备用内核文件可以在标准操作系统内核源码的基础上，减少系统恢复时必须检索以及依赖的数据结构，从而可以加快内核的启动速度，减少系统恢复过程中的耗时。此外，减少系统恢复时必须检索以及依赖的数据结构，也能减少内核代码对内存资源的占用。

[0070] 本申请实施例提供一种内核崩溃恢复方法，图3A为本申请实施例提供的内核崩溃恢复方法的实现流程示意图，如图3A所示，该方法包括：

[0071] 步骤S301，处理器上电后，引导程序基于主内核文件，加载并初始化主内核；

[0072] 步骤S302，所述主内核基于备用内核文件，在所述第一内存区域中加载备用内核；其中，所述主内核文件与所述备用内核文件中的内存偏移量不同；

[0073] 步骤S303，当主内核崩溃时，所述主内核控制处理器保存正在执行的正常线程的上下文；

[0074] 步骤S304，当所述正常线程的上下文保存完成后，所述主内核在第一内存区域中初始化已加载的备用内核；其中，所述第一内存区域为所述主内核在物理内存中预留的内存区域；

[0075] 步骤S305，当初始化完成后，所述备用内核通过访问所述主内核的内核数据结构，获得所述正常线程的上下文；

[0076] 步骤S306,所述备用内核根据所述正常线程的上下文,恢复全部进程的运行状态;

[0077] 这里,需要说明的是,步骤S301至S306可以采用与前述步骤S201至S206同样的方式实施,在此不再赘述。

[0078] 步骤S307,当所述全部进程的运行状态恢复后,所述备用内核作为新的主内核运行,回收可用的物理内存;

[0079] 这里,在恢复所有待恢复的应用程序进程之后,之前加载的备用内核已变成新的主内核运行,新的主内核会回收所有可用的物理内存资源,并将回收的物理内存资源添加到空闲内存列表中,以供后续运行时的内存资源分配。

[0080] 步骤S308,所述新的主内核在物理内存中预留第二内存区域,并轮流基于所述主内核文件和所述备用内核文件二者之一,在所述第二内存区域中加载新的备用内核;其中,所述新的主内核对应的内核文件与所述新的备用内核对应的内核文件不同。

[0081] 这里,在每次恢复的过程中,新的备用内核总是基于存储在磁盘上的主内核文件和备用内核文件当中,与所述新的备用内核对应的内核文件不同的那一个内核文件,加载新的备用内核。这样随着主备内核的相继更替,新的主内核会轮流基于所述主内核文件和所述备用内核文件二者之一,来加载新的备用内核。例如,在系统第一次启动时,主内核基于主内核文件加载并初始化,备用内核基于备用内核文件加载;当主内核崩溃后,备用内核作为新的主内核运行,此时,新的主内核会基于主内核文件加载新的备用内核;当下一次主内核崩溃后,备用内核作为新的主内核运行,此时,新的主内核会基于备用内核文件加载新的备用内核;这样,随着主内核的每次崩溃,新的主内核会轮流基于存储在磁盘上的主内核文件和备用内核文件中的一个内核文件,加载新的备用内核文件。

[0082] 新的备用内核加载完成后,说明此次内核崩溃已经恢复完成,旧的备用内核已经变成主内核运行,而新加载的备用内核会作为下一次内核崩溃时的备用内核。图3B为恢复完成后的系统运行状态示意图,如图3B所示,此时前备用内核112、硬件121以及进程131、132、133均处于正常运行状态,新的备用内核113处于未运行状态。

[0083] 在实施的时候,新的主内核加载新的备用内核的方法,可由本领域技术人员根据实际场景自由选择,本申请实施例对此并不做限定。例如,当所述主内核文件与所述备用内核文件为Linux内核文件时,所述新的主内核可以基于所述主内核文件或所述备用内核文件,利用kdump机制加载新的备用内核。

[0084] 在一些实施例中,在加载完备用内核后,主内核还可以对第二内存区域进行内存保护,以避免加载的备用内核被其他进程修改。在实施时,可根据实际需求,对第二内存区域的保护属性进行设置,包括但不限于将保护属性设置为不可写且不可执行,和/或不可访问。相应地,主内核崩溃时,当所述正常线程的上下文保存完成后,主内核需要将第二内存区域的内存保护解除,使得主内核可以访问到备用内核的初始化入口点,且备用内核的初始化代码可执行。在实施时,主内核可以对第二内存区域的保护属性进行设置,包括但不限于将保护属性设置为可写可执行,和/或可访问。

[0085] 在实施的时候,预留的第二内存区域的大小可以根据内核加载及初始化时实际需要的内存大小设置,本实施例对此不作限定。

[0086] 本申请实施例提供一种内核崩溃恢复方法,图4为本申请实施例提供的内核崩溃恢复方法的实现流程示意图,如图4所示,该方法包括:

[0087] 步骤S401,当主内核崩溃时,所述主内核通过错误处理机制,控制由主内核崩溃导致崩溃的处理器向正常处理器发送不可屏蔽中断;

[0088] 这里,主内核崩溃时,执行发生崩溃的线程的处理器也会随之崩溃,触发错误处理机制。此时,其他处理器还能正常执行线程。

[0089] 在实施时,可以通过对操作系统默认的崩溃处理机制进行修改,在主内核崩溃时,不进行重新引导,而是执行自定义的崩溃处理程序,向所有正常处理器发出不可屏蔽中断。

[0090] 步骤S402,所述主内核控制每一所述正常处理器接收所述不可屏蔽中断后,保存正在执行的线程的上下文;

[0091] 这里,在接收到不可屏蔽的中断后,每个正常处理器保存当前正在执行的线程的上下文,将线程的中央处理器(CPU)寄存器中的值都保存在相应的内核堆栈上。

[0092] 步骤S403,所述主内核控制每一所述正常处理器在保存完正在执行的线程的上下文后停止运行;

[0093] 在实施时,每个正常处理器在保存完当前正在执行的线程的上下文后,可以设置一个全局标志,指示已经完成上下文保存,并使自己停止,这样可以确保将控制权交给备用内核时,所有用户线程的CPU寄存器中的值都已保存在相应的内核堆栈上。

[0094] 步骤S404,所述主内核检测到全部正常处理器停止后,控制所述崩溃的处理器执行所述备用内核的初始化程序,在第一内存区域中初始化已加载的备用内核;

[0095] 这里,主内核可以在启动时预留第一内存区域,所述主内核在启动后将备用内核加载至第一内存区域,但不做初始化。只要主内核运行正常,备用内核就会保留在第一内存区域中,并且它的代码永远不会执行。

[0096] 在实施时,主内核检测全部正常处理器是否停止的方式,本申请实施例并不限定,本领域技术人员可以根据实际情况自由选择。在一些实施例中,主内核可以通过读取每个正常处理器在保存完当前正在执行的线程的上下文后设置的全局变量,判断各个正常处理器是否停止运行。当全部正常处理器停止后,主内核控制崩溃的处理器跳转到备用内核的初始化点,开始执行备用内核的初始化,此时,系统不需要重启,备用内核开始在预留的第一内存区域中初始化,主内核不再执行任何代码。

[0097] 在一些实施例中,为防止加载的备用内核被其他进程修改,主内核会对备用内核加载处所在的第一内存区域进行内存保护,在崩溃的处理器跳转到备用内核的初始化点之前,需要将该第一内存区域的内存保护解除,使得主内核可以访问到备用内核的初始化入口点,且备用内核的初始化代码可执行。在实施时,主内核可以对第一内存区域的保护属性进行设置,包括但不限于将保护属性设置为可写可执行,和/或可访问。

[0098] 此外,在备用内核初始化时,可以通过修改启动代码中的内核启动参数来动态修改所述备用内核的可用物理内存大小。在一些实施例中,备用内核的启动代码中还需要分配额外的内存页面描述符,这些描述符会在备用内核恢复过程中使用。在一些实施例中,为了不破坏主内核交换出的任何页面,可以在系统中划分两个交换分区:一个由主内核使用,另一个由备用内核使用。内核启动时使用的交换分区可以由系统启动脚本根据内核版本进行选择。在实施时,可以将内核版本区分为主内核和备用内核两种版本,主内核崩溃后,备用内核在恢复过程中会根据主内核的交换分区中的数据在备用内核对应的交换分区中重新生成一份对应的新的数据。

[0099] 步骤S405,当初始化完成后,所述备用内核通过访问所述主内核的内核数据结构,获得所述正常线程的上下文;

[0100] 步骤S406,所述备用内核根据所述正常线程的上下文,恢复全部进程的运行状态。

[0101] 这里,需要说明的是,步骤S405至S406可以采用与前述步骤S103至S104同样的方式实施,在此不再赘述。

[0102] 本申请实施例提供一种内核崩溃恢复方法,图5为本申请实施例提供的内核崩溃恢复方法的实现流程示意图,如图5所示,该方法包括:

[0103] 步骤S501,当主内核崩溃时,所述主内核控制处理器保存正在执行的正常线程的上下文;

[0104] 步骤S502,当所述正常线程的上下文保存完成后,所述主内核在第一内存区域中初始化已加载的备用内核;其中,所述第一内存区域为所述主内核在物理内存中预留的内存区域;

[0105] 这里,步骤S501至S502可以采用与前述步骤S101至S102同样的方式实施,在此不再赘述。

[0106] 步骤S503,当初始化完成后,所述备用内核通过访问所述主内核的内核数据结构,获得所述正常线程的上下文;其中,所述正常线程的上下文包括所述主内核的进程描述符链表、内存区域描述符表、虚拟内存页、文件描述符表;

[0107] 备用内核在完成初始化之后,将开始恢复阶段。在恢复阶段,备用内核将访问主内核的内核数据结构,获得主内核崩溃时保存的正常线程的上下文。

[0108] 步骤S504,所述备用内核访问所述主内核的进程描述符链表,读取待恢复的进程列表;

[0109] 这里,在主内核崩溃时,通过保存正常线程的上下文,可以将崩溃时运行的全部进程保存在主内核的进程描述符链表中。备用内核可以通过访问该链表,读取到全部待恢复的进程。在实施时,以Linux内核为例,进程描述符被放在一个链表中,这个链表的第一个元素的位置存储在内核中的一个全局变量中,由于内核的起始物理地址是常量,并且在内核编译时是可配置的,因此,备用内核可以通过主内核的起始物理地址常量和该全局变量,确定进程描述符链表中第一个元素的物理地址,从而访问该链表中的每一个进程描述符,获得待恢复的每个进程。

[0110] 步骤S505,所述备用内核为所述待恢复的进程列表中的每一待恢复进程,创建一个新进程;

[0111] 这里,对于要恢复的每个进程,备用内核都会创建一个新进程,新创建的进程的虚拟地址空间的内核部分与在备用内核上运行的任何其他进程的虚拟地址空间的内核部分相同。

[0112] 在实施时,备用内核可以默认恢复全部进程。在一些实施例中,可以在屏幕上为用户提供选择需要恢复的进程的操作页面,让用户选择需要恢复的进程。在一些实施例中,还可以通过生成配置文件写入需要恢复的进程,启动脚本可以读取配置文件来恢复对应的进程,从而实现在无人值守时恢复。

[0113] 步骤S506,所述备用内核根据所述主内核的内存区域描述符列表的内容,在每一所述新进程中分别恢复对应的待恢复进程的用户内存空间;

[0114] 这里，要将每一个新创建的进程的虚拟地址空间的用户部分恢复为对应的正在恢复的进程的虚拟地址空间的用户部分的副本。在实施时，对于每一所述新进程，备用内核可以从主内核的内存数据结构中获取对应的要恢复的进程的内存区域描述符列表；根据列表中的每个内存描述符，备用内核可以为所述新进程创建一个具有相同属性的新内存描述符。

[0115] 在一些实施例中，当根据内存描述符确定对应内存中存在内存映射文件时，所述备用内核可以在与所述内存区域描述符对应的新进程中打开所述文件，并将所述文件重新映射到所述备用内核中相应的内存区域。

[0116] 在一些实施例中，所述正常线程的上下文还包括所述主内核的交换区域描述符。当所述文件映射的内存区域在交换分区时，所述备用内核检索所述主内核的交换区域描述符，获得指向与所述文件对应的文件结构的指针；所述备用内核通过所述文件结构的指针，获得所述文件结构的内容；所述备用内核根据所述文件结构的内容，将所述文件重新打开，并将所述文件重新映射到所述备用内核中相应的内存区域。在实施时，所述备用内核可以根据文件结构中的文件名或设备的符号名称，将对应的文件重新打开。

[0117] 步骤S507，所述备用内核根据所述主内核的每个虚拟内存页的内容，在每一所述新进程中分别恢复对应的待恢复进程的硬件页表和交换页表；

[0118] 在实施时，虚拟内存页包括但不限于每一所述待恢复进程的硬件页表和交换页表。当所述备用内核从所述待恢复进程的硬件页表中检索到存在条目时，所述备用内核为每一所述条目在所述备用内核中分配一个新页，并将每一所述条目相应页的内容分别复制到对应的新页。所述备用内核为所述主内核交换到磁盘的每个页面，在所述备用内核的交换分区中分配一个新页，并将所述主内核交换到磁盘的每个页面的内容分别复制到对应的新页。在硬件页表和交换页表恢复后，每个待恢复进程的用户内存空间已完全恢复。

[0119] 步骤S508，所述备用内核根据所述主内核的文件描述符表的内容，在每一所述新进程中恢复打开的文件。

[0120] 这里，在每一所述新进程中恢复打开的文件，包括重新打开待恢复进程在主内核崩溃时打开的文件。在一些实施例中，还可以将文件恢复至与主内核中相同的位置，并恢复当前偏移量。在实施时，所述备用内核可以访问所述主内核的文件描述符表，读取每一所述待恢复的进程中文件的名称、位置、打开标志和当前偏移量；所述备用内核根据每一所述待恢复的进程中文件的名称、位置、打开标志和当前偏移量，分别在对应的所述新进程中恢复所述文件的打开状态。

[0121] 在一些实施例中，备用内核还可以将主内核内存中的文件数据置为脏的缓存页回刷到磁盘，从而将主内核崩溃时未保存至磁盘的文件更改保存至磁盘，实现全部数据的恢复。

[0122] 本申请实施例提供一种内核崩溃恢复方法，图6为本申请实施例提供的内核崩溃恢复方法的实现流程示意图，如图6所示，该方法包括：

[0123] 步骤S601，系统启动；

[0124] 这里，在系统启动前，需要先编译内存偏移量不同的主内核文件和备用内核文件。可以对操作系统标准内核进行修改，减少系统恢复时必须检索以及依赖的数据结构，编译出一样的主内核和备用内核，不同之处在于两个内核的内存偏移量不同。

[0125] 步骤S602,加载主备内核;

[0126] 这里,操作系统启动时,引导程序先加载并初始化主内核,主内核启动后预留出一块物理内存,利用kdump加载备用内核,并做内存保护,但不做初始化。只要主内核在没有失败的情况下运行,备用内核就会保留在物理内存的这个区域中,并且它的代码永远不会执行。在实施时,物理内存的这个区域可以是主内核预留的第一内存区域。

[0127] 步骤S603,系统正常运行;

[0128] 步骤S604,内核崩溃;

[0129] 这里,运行的内核为主内核,当主内核发生严重错误时,例如crash,崩溃处理程序不会重新引导,而是向所有处理器发出不可屏蔽中断,发生故障的CPU除外。在实施时,故障CPU可以通过执行崩溃处理代码向所有正常处理器发送不可屏蔽中断,使其保存当前线程上下文。在接收到不可屏蔽的中断后,每个处理器保存它正在执行的线程的上下文,然后设置一个全局标志,指示已经保存了上下文,并使自己停止。这确保了将控制权交给备用内核时,所有用户线程的CPU寄存器都保存在相应的内核堆栈上。后续备用内核在恢复时会检索这个上下文,并像常规上下文切换一样继续执行线程。正在执行故障代码的处理器等待所有其他处理器停止后,将备用内核镜像加载处的内存保护删除,并跳转到备用内核的初始化点,开始执行备用内核的初始化。这里备用内核镜像加载处在实施时可以是主内核预留的第一内存区域。

[0130] 步骤S605,切换备用内核;

[0131] 这里,备用内核开始在预留的内存中初始化,备用内核通过修改内核参数来动态修改可用内存大小,主内核不再执行任何代码,此时,系统不需要重启。

[0132] 在实施时,为了能够动态地更改可用物理内存的数量,需要修改Linux的启动代码中的内核启动参数,备用内核的启动代码中必须分配额外的内存页面描述符,这些描述符会在备用内核恢复过程完成时使用。同时为了不破坏主内核交换出的任何页面,在系统中划分两个交换分区:一个由主内核使用,另一个由备用内核使用。内核启动时使用的交换分区可以由系统启动脚本根据内核版本进行选择。

[0133] 步骤S606,恢复系统;

[0134] 这里,在备用内核完成初始化之后,将开始恢复阶段。在此阶段,备用内核将访问主内核的内核数据结构,以便恢复应用程序。默认情况下恢复全部进程,当然也可以在屏幕上选择需要恢复的进程,或者生成配置文件写入需要恢复的进程,启动脚本会读取配置文件来恢复对应的进程,可以在无人值守时恢复。

[0135] 恢复过程可以包括以下步骤:

[0136] 第一步,恢复进程描述符,在Linux中,进程描述符被放在一个链表中。这个链表的第一个元素的位置存储在内核中的一个全局变量中。由于内核的起始物理地址是常量,并且在内核编译时是可配置的,因此,备用内核可以通过主内核的起始物理地址常量和该全局变量,确定进程描述符链表中第一个元素的物理地址,从而访问该链表中的每一个进程描述符,获得待恢复的每个进程。

[0137] 第二步,备用内核从主内核检索交换区域描述符,该描述符存储在一个固定大小的数组中,可以通过另一个全局变量访问。每个描述符描述一个交换分区,并包含一个指向文件结构的指针,该文件结构对应于一个常规文件或存储交换区域的设备文件。由于设备

的符号名称存储在这个结构中，备用内核可以重新打开对应的文件。对于要恢复的每个进程，备用内核都会创建一个新进程。新创建进程的虚拟地址空间的内核部分与在备用内核上运行的任何其他进程相同。新创建进程的虚拟地址空间的用户部分为正在恢复的进程的虚拟地址空间的用户部分的副本。为此，备用内核需从主内核的内存数据结构中获取要恢复的进程的内存区域描述符列表。对于列表中的每个内存描述符，备用内核将创建一个具有相同属性的新内存描述符。如果内存中有文件映射，备用内核会将这些文件重新打开并映射到相应的内存区域。

[0138] 第三步，检索内存区域内每个虚拟内存页的内容。备用内核从主内核的内存数据结构中检索正在恢复的进程的硬件页表的对应条目，如果有，则在备用内核中分配一个新页，并将主内核相应页的内容复制到其中。对于主内核交换到磁盘的页面的每个条目，备用内核都会在备用内核的交换分区中分配一个新页。这将完全恢复每个要恢复的进程的用户内存空间。

[0139] 第四步，恢复进程打开的文件。备用内核从主内核的文件描述符表读取名称、位置、打开标志和当前偏移量，并相应地重新打开文件，将文件放在与主内核中相同的位置，并恢复当前偏移量，最后，将主内核内存中的文件数据置为脏的缓存页回刷到磁盘，从而将主内核崩溃时未保存至磁盘的文件更改保存至磁盘，实现全部数据的恢复。

[0140] 步骤S607，加载新备用内核。

[0141] 这里，备用内核恢复全部进程后，作为新的主内核继续运行，新的主内核可以利用kdump机制加载新的备用内核。在恢复所有目标应用程序进程之后，备用内核会回收所有可用的物理内存，将其添加到空闲内存列表中，并重新分配一块内存作为内存保护区域，利用kdump将磁盘中编译好的主内核加载进去，此时之前加载的备用内核已变成主内核运行，而新加载的主内核会作为下一次内核错误的备用内核。

[0142] 基于前述的实施例，本申请实施例提供一种内核崩溃恢复装置，该装置包括所包括的各单元、以及各单元所包括的各模块，可以通过计算机设备中的处理器来实现；当然也可通过具体的逻辑电路实现；在实施的过程中，处理器可以为中央处理器(CPU)、微处理器(MPU)、数字信号处理器(DSP)或现场可编程门阵列(FPGA)等。

[0143] 图7为本申请实施例内核崩溃恢复装置的组成结构示意图，如图7所示，所述装置700包括主内核701和备用内核702，其中：

[0144] 所述主内核701，用于：当主内核崩溃时，控制处理器保存正在执行的正常线程的上下文；当所述正常线程的上下文保存完成后，在第一内存区域中初始化已加载的备用内核；其中，所述第一内存区域为所述主内核在物理内存中预留的内存区域；

[0145] 所述备用内核702，用于：当初始化完成后，通过访问所述主内核的内核数据结构，获得所述正常线程的上下文；根据所述正常线程的上下文，恢复全部进程的运行状态。

[0146] 在一些实施例中，所述系统还包括引导程序，用于处理器上电后，基于主内核文件，加载并初始化主内核。所述主内核还用于基于备用内核文件，在所述第一内存区域中加载备用内核；其中，所述主内核文件与所述备用内核文件中的内存偏移量不同。

[0147] 在一些实施例中，所述备用内核还用于：当所述全部进程的运行状态恢复后，作为新的主内核运行，回收可用的物理内存；在物理内存中预留第二内存区域，并轮流基于所述主内核文件和所述备用内核文件二者之一，在所述第二内存区域中加载新的备用内核；其

中，所述新的主内核对应的内核文件与所述新的备用内核对应的内核文件不同。

[0148] 在一些实施例中，所述主内核还用于：当主内核崩溃时，通过错误处理机制，控制崩溃的处理器向正常处理器发送不可屏蔽中断；控制每一所述正常处理器接收所述不可屏蔽中断后，保存正在执行的线程的上下文；控制每一所述正常处理器在保存完正在执行的线程的上下文后停止运行；检测到全部正常处理器停止后，控制所述崩溃的处理器执行所述备用内核的初始化程序，在第一内存区域中初始化已加载的备用内核。

[0149] 在一些实施例中，所述备用内核还用于：访问所述主内核的进程描述符链表，读取待恢复的进程列表；为所述待恢复的进程列表中的每一待恢复进程，创建一个新进程；根据所述主内核的内存区域描述符列表的内容，在每一所述新进程中分别恢复对应的待恢复进程的用户内存空间；根据所述主内核的每个虚拟内存页的内容，在每一所述新进程中分别恢复对应的待恢复进程的硬件页表和交换页表；根据所述主内核的文件描述符表的内容，在每一所述新进程中恢复打开的文件。

[0150] 在一些实施例中，所述备用内核还用于：访问所述主内核中每一所述待恢复进程的内存区域描述符列表；根据每一所述待恢复进程的内存区域描述符列表中的每一内存区域描述符，分别为对应的所述新进程创建一个具有相同属性的新内存描述符。

[0151] 在一些实施例中，所述备用内核还用于：当所述内存区域描述符中存在内存映射文件时，在与所述内存区域描述符对应的新进程中打开所述文件；将所述文件重新映射到所述备用内核中相应的内存区域。

[0152] 在一些实施例中，所述备用内核还用于：当所述文件映射的内存区域在交换分区时，检索所述主内核的交换区域描述符，获得指向与所述文件对应的文件结构的指针；通过所述文件结构的指针，获得所述文件结构的内容；根据所述文件结构的内容，将所述文件重新打开，并将所述文件重新映射到所述备用内核中相应的内存区域。

[0153] 在一些实施例中，所述备用内核还用于：当所述备用内核从所述待恢复进程的硬件页表中检索到存在条目时，为每一所述条目在所述备用内核中分配一个新页，并将每一所述条目相应页的内容分别复制到对应的新页；为所述主内核交换到磁盘的每个页面，在所述备用内核的交换分区中分配一个新页，并将所述主内核交换到磁盘的每个页面的内容分别复制到对应的新页。

[0154] 在一些实施例中，所述备用内核还用于：访问所述主内核的文件描述符表，读取每一所述待恢复的进程中文件的名称、位置、打开标志和当前偏移量；根据每一所述待恢复的进程中文件的名称、位置、打开标志和当前偏移量，分别在对应的所述新进程中恢复所述文件的打开状态。

[0155] 以上装置实施例的描述，与上述方法实施例的描述是类似的，具有同方法实施例相似的有益效果。对于本申请装置实施例中未披露的技术细节，请参照本申请方法实施例的描述而理解。

[0156] 需要说明的是，本申请实施例中，如果以软件功能模块的形式实现上述的内核崩溃恢复方法，并作为独立的产品销售或使用时，也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相关技术做出贡献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)执行本申请各

个实施例所述方法的全部或部分。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)、磁碟或者光盘等各种可以存储程序代码的介质。这样,本申请实施例不限制于任何特定的硬件和软件结合。

[0157] 对应地,本申请实施例提供一种计算机可读存储介质,其上存储有计算机程序,该计算机程序被处理器执行时实现上述方法中的步骤。

[0158] 对应地,本申请实施例提供一种计算机设备,包括存储器和处理器,所述存储器存储有可在处理器上运行的计算机程序,所述处理器执行所述程序时实现上述方法中的步骤。

[0159] 这里需要指出的是:以上存储介质和设备实施例的描述,与上述方法实施例的描述是类似的,具有同方法实施例相似的有益效果。对于本申请存储介质和设备实施例中未披露的技术细节,请参照本申请方法实施例的描述而理解。

[0160] 需要说明的是,图8为本申请实施例中计算机设备的一种硬件实体示意图,如图8所示,该计算机设备800的硬件实体包括:处理器801、通信接口802和存储器803。

[0161] 处理器801通常控制计算机设备800的总体操作。

[0162] 通信接口802可以使计算机设备通过网络与其他终端或服务器通信。

[0163] 存储器803配置为存储由处理器801可执行的指令和应用,还可以缓存待处理器801以及计算机设备800中各模块待处理或已经处理的数据(例如,图像数据、音频数据、语音通信数据和视频通信数据),可以通过闪存(FLASH)或随机访问存储器(Random Access Memory,RAM)实现。

[0164] 应理解,说明书通篇中提到的“一个实施例”或“一实施例”意味着与实施例有关的特定特征、结构或特性包括在本申请的至少一个实施例中。因此,在整个说明书各处出现的“在一个实施例中”或“在一实施例中”未必一定指相同的实施例。此外,这些特定的特征、结构或特性可以任意适合的方式结合在一个或多个实施例中。应理解,在本申请的各种实施例中,上述各过程的序号的大小并不意味着执行顺序的先后,各过程的执行顺序应以其功能和内在逻辑确定,而不应对本申请实施例的实施过程构成任何限定。上述本发明实施例序号仅仅为了描述,不代表实施例的优劣。

[0165] 需要说明的是,在本文中,术语“包括”、“包含”或者其任何其他变体意在涵盖非排他性的包含,从而使得包括一系列要素的过程、方法、物品或者装置不仅包括那些要素,而且还包括没有明确列出的其他要素,或者是还包括为这种过程、方法、物品或者装置所固有的要素。在没有更多限制的情况下,由语句“包括一个……”限定的要素,并不排除在包括该要素的过程、方法、物品或者装置中还存在另外的相同要素。

[0166] 在本申请所提供的几个实施例中,应该理解到,所揭露的设备和方法,可以通过其它的方式实现。以上所描述的设备实施例仅仅是示意性的,例如,所述单元的划分,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,如:多个单元或组件可以结合,或可以集成到另一个系统,或一些特征可以忽略,或不执行。另外,所显示或讨论的各组成部分相互之间的耦合、或直接耦合、或通信连接可以是通过一些接口,设备或单元的间接耦合或通信连接,可以是电性的、机械的或其它形式的。

[0167] 上述作为分离部件说明的单元可以是、或也可以不是物理上分开的,作为单元显示的部件可以是、或也可以不是物理单元;既可以位于一个地方,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全部单元来实现本实施例方案的目的。

[0168] 另外,在本发明各实施例中的各功能单元可以全部集成在一个处理单元中,也可以是各单元分别单独作为一个单元,也可以两个或两个以上单元集成在一个单元中;上述集成的单元既可以采用硬件的形式实现,也可以采用硬件加软件功能单元的形式实现。

[0169] 本领域普通技术人员可以理解:实现上述方法实施例的全部或部分步骤可以通过程序指令相关的硬件来完成,前述的程序可以存储于计算机可读取存储介质中,该程序在执行时,执行包括上述方法实施例的步骤;而前述的存储介质包括:移动存储设备、只读存储器(Read Only Memory,ROM)、磁碟或者光盘等各种可以存储程序代码的介质。

[0170] 或者,本发明上述集成的单元如果以软件功能模块的形式实现并作为独立的产品销售或使用时,也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,本发明实施例的技术方案本质上或者说对相关技术做出贡献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)执行本发明各个实施例所述方法的全部或部分。而前述的存储介质包括:移动存储设备、ROM、磁碟或者光盘等各种可以存储程序代码的介质。

[0171] 以上所述,仅为本发明的实施方式,但本发明的保护范围并不局限于此,任何熟悉本技术领域的技术人员在本发明揭露的技术范围内,可轻易想到变化或替换,都应涵盖在本发明的保护范围之内。因此,本发明的保护范围应以所述权利要求的保护范围为准。

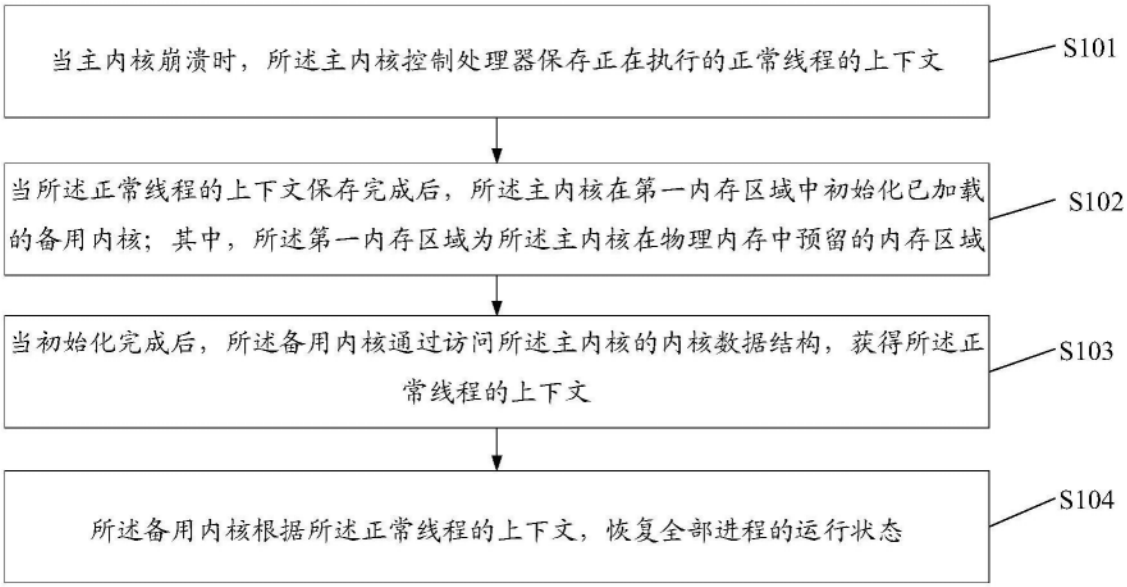

当主内核崩溃时，所述主内核控制处理器保存正在执行的正常线程的上下文
S101
当所述正常线程的上下文保存完成后，所述主内核在第一内存区域中初始化已加载的备用内核；其中，所述第一内存区域为所述主内核在物理内存中预留的内存区域
S102
当初始化完成后，所述备用内核通过访问所述主内核的内核数据结构，获得所述正常线程的上下文
S103
所述备用内核根据所述正常线程的上下文，恢复全部进程的运行状态
S104

图1A


进程131
进程132
进程133
主内核111
备用内核112
硬件121

图1B


进程131
进程132
进程133
崩溃
主内核111
备用内核112
硬件121

图1C

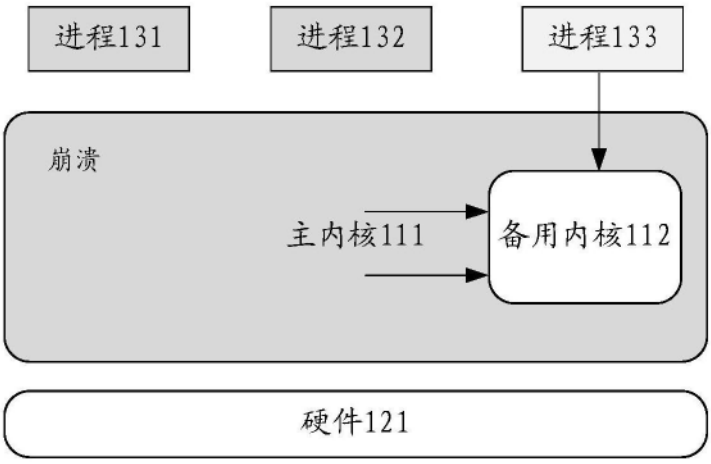

进程131
进程132
进程133
崩溃
主内核111
备用内核112
硬件121

图1D


处理器上电后，引导程序基于主内核文件，加载并初始化主内核
S201
所述主内核基于备用内核文件，在所述第一内存区域中加载备用内核；其中，所述主内核文件与所述备用内核文件中的内存偏移量不同
S202
当主内核崩溃时，所述主内核控制处理器保存正在执行的正常线程的上下文
S203
当所述正常线程的上下文保存完成后，所述主内核在第一内存区域中初始化已加载的备用内核；其中，所述第一内存区域为所述主内核在物理内存中预留的内存区域
S204
当初始化完成后，所述备用内核通过访问所述主内核的内核数据结构，获得所述正常线程的上下文
S205
所述备用内核根据所述正常线程的上下文，恢复全部进程的运行状态
S206

图2

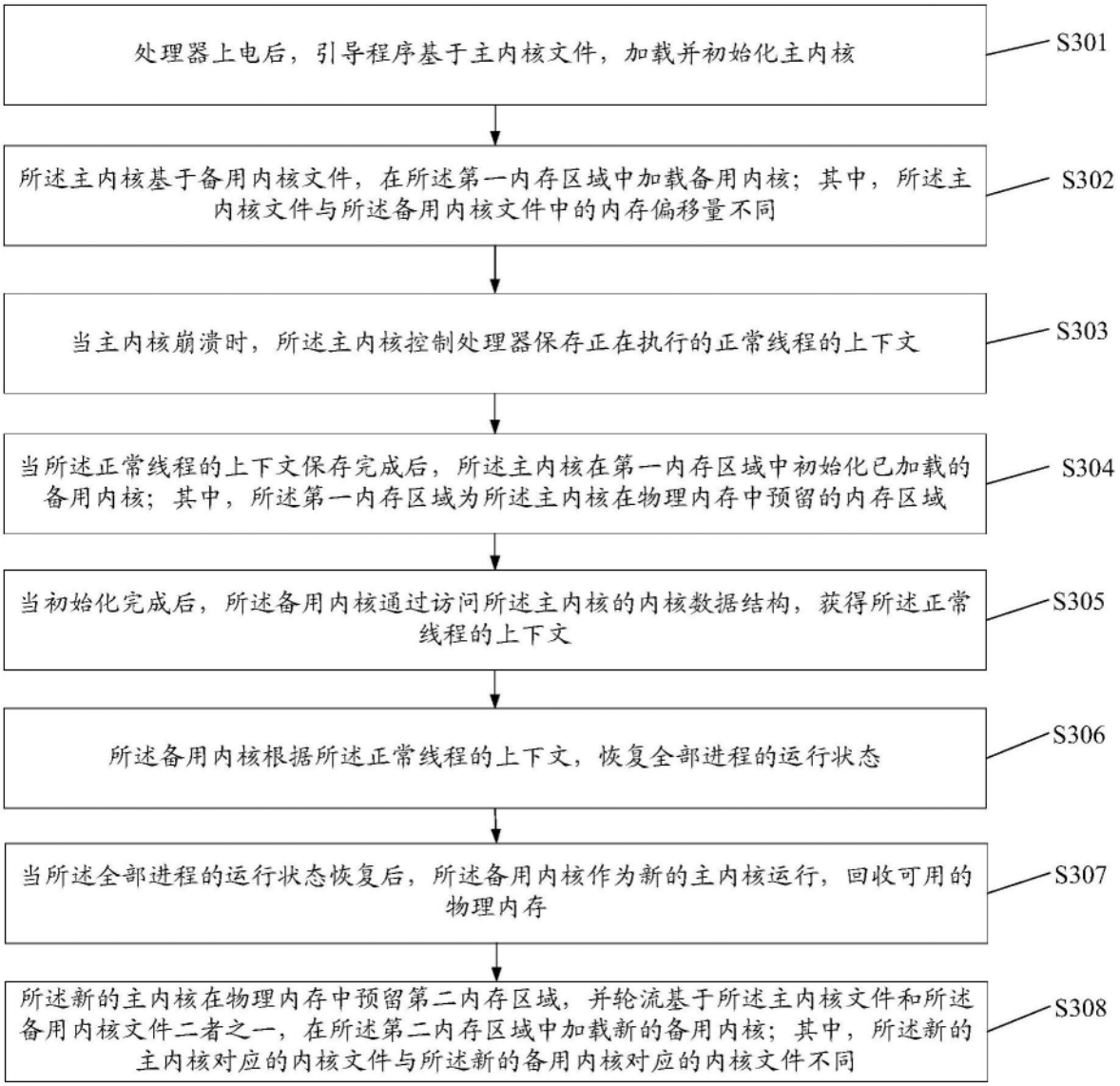

处理器上电后，引导程序基于主内核文件，加载并初始化主内核
S301
所述主内核基于备用内核文件，在所述第一内存区域中加载备用内核；其中，所述主内核文件与所述备用内核文件中的内存偏移量不同
S302
当主内核崩溃时，所述主内核控制处理器保存正在执行的正常线程的上下文
S303
当所述正常线程的上下文保存完成后，所述主内核在第一内存区域中初始化已加载的备用内核；其中，所述第一内存区域为所述主内核在物理内存中预留的内存区域
S304
当初始化完成后，所述备用内核通过访问所述主内核的内核数据结构，获得所述正常线程的上下文
S305
所述备用内核根据所述正常线程的上下文，恢复全部进程的运行状态
S306
当所述全部进程的运行状态恢复后，所述备用内核作为新的主内核运行，回收可用的物理内存
S307
所述新的主内核在物理内存中预留第二内存区域，并轮流基于所述主内核文件和所述备用内核文件二者之一，在所述第二内存区域中加载新的备用内核；其中，所述新的主内核对应的内核文件与所述新的备用内核对应的内核文件不同
S308

图3A

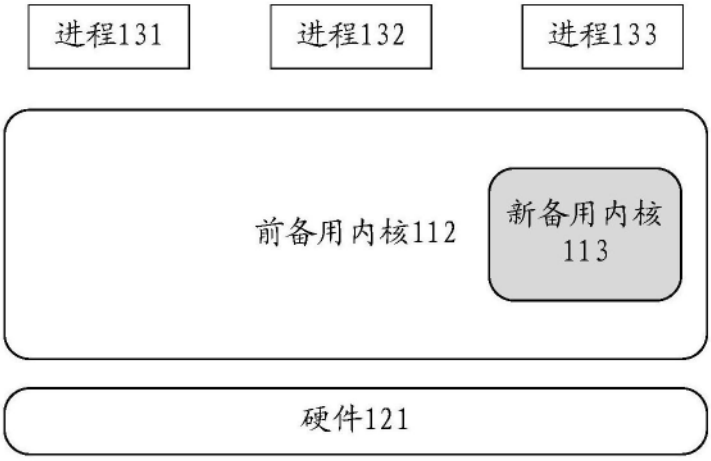

进程131
进程132
进程133
前备用内核112
新备用内核113
硬件121

图3B


当主内核崩溃时，所述主内核通过错误处理机制，控制由主内核崩溃导致崩溃的处理器向正常处理器发送不可屏蔽中断
S401
所述主内核控制每一所述正常处理器接收所述不可屏蔽中断后，保存正在执行的线程的上下文
S402
所述主内核控制每一所述正常处理器在保存完正在执行的线程的上下文后停止运行
S403
所述主内核检测到全部正常处理器停止后，控制所述崩溃的处理器执行所述备用内核的初始化程序，在第一内存区域中初始化已加载的备用内核
S404
当初始化完成后，所述备用内核通过访问所述主内核的内核数据结构，获得所述正常线程的上下文
S405
所述备用内核根据所述正常线程的上下文，恢复全部进程的运行状态
S406

图4

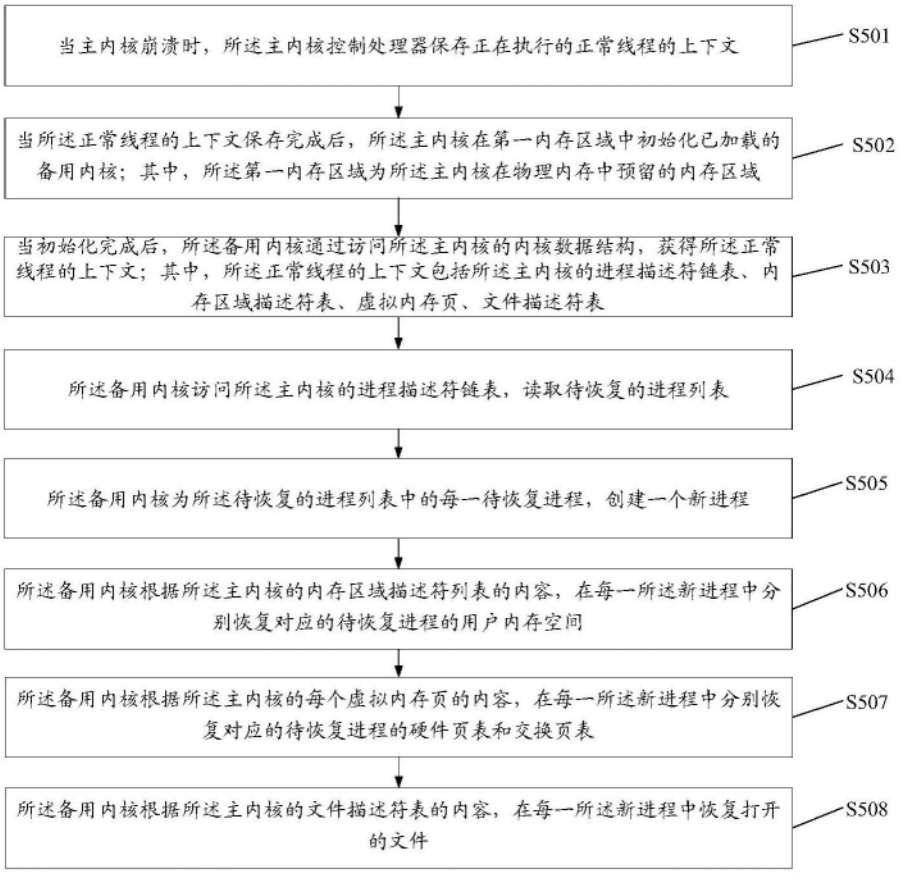

当主内核崩溃时，所述主内核控制处理器保存正在执行的正常线程的上下文
S501
当所述正常线程的上下文保存完成后，所述主内核在第一内存区域中初始化已加载的备用内核；其中，所述第一内存区域为所述主内核在物理内存中预留的内存区域
S502
当初始化完成后，所述备用内核通过访问所述主内核的内核数据结构，获得所述正常线程的上下文；其中，所述正常线程的上下文包括所述主内核的进程描述符链表、内存区域描述符表、虚拟内存页、文件描述符表
S503
所述备用内核访问所述主内核的进程描述符链表，读取待恢复的进程列表
S504
所述备用内核为所述待恢复的进程列表中的每一待恢复进程，创建一个新进程
S505
所述备用内核根据所述主内核的内存区域描述符列表的内容，在每一所述新进程中分别恢复对应的待恢复进程的用户内存空间
S506
所述备用内核根据所述主内核的每个虚拟内存页的内容，在每一所述新进程中分别恢复对应的待恢复进程的硬件页表和交换页表
S507
所述备用内核根据所述主内核的文件描述符表的内容，在每一所述新进程中恢复打开的文件
S508

图5

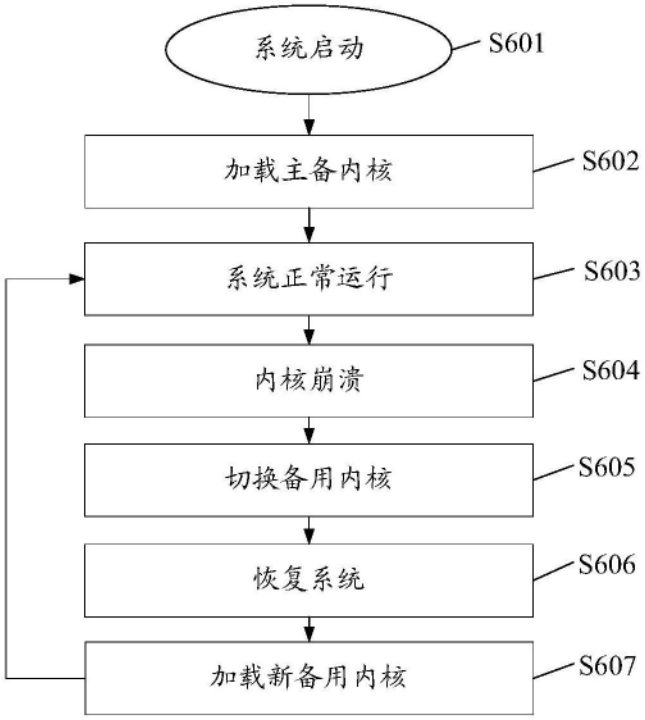

系统启动
S601
加载主备内核
S602
系统正常运行
S603
内核崩溃
S604
切换备用内核
S605
恢复系统
S606
加载新备用内核
S607

图6

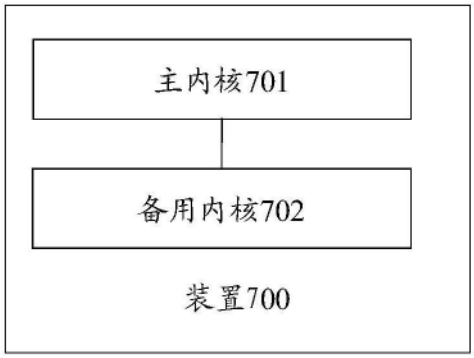

主内核701
备用内核702
装置700

图7

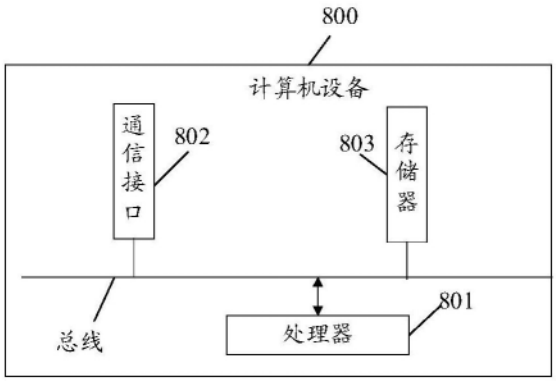


图8