(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 104508626 B
(45)授权公告日 2017.06.13

(21)申请号 201380039991.8

(22)申请日 2013.07.26

(65)同一申请的已公布的文献号
申请公布号 CN 104508626 A

(43)申请公布日 2015.04.08

(30)优先权数据
13/560,216 2012.07.27 US

(85)PCT国际申请进入国家阶段日
2015.01.27

(86)PCT国际申请的申请数据
PCT/US2013/052153 2013.07.26

(87)PCT国际申请的公布数据
WO2014/018812 EN 2014.01.30

(73)专利权人 微软公司
地址 美国华盛顿州

(72)发明人 K·雷厄森

(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 潘明婭

(51)Int.Cl.
G06F 9/30(2006.01)
G06F 9/38(2006.01)

(56)对比文件
US 2003/0023960 A1,2003.01.30,
US 2003/0023960 A1,2003.01.30,
CN 101233495 A,2008.07.30,
US 2004/0059641 A1,2004.03.25,
US 2009/0307528 A1,2009.12.10,
CN 101853148 A,2010.10.06,
US 6253309 B1,2001.06.26,
US 7581082 B2,2009.08.25,

审查员 张力

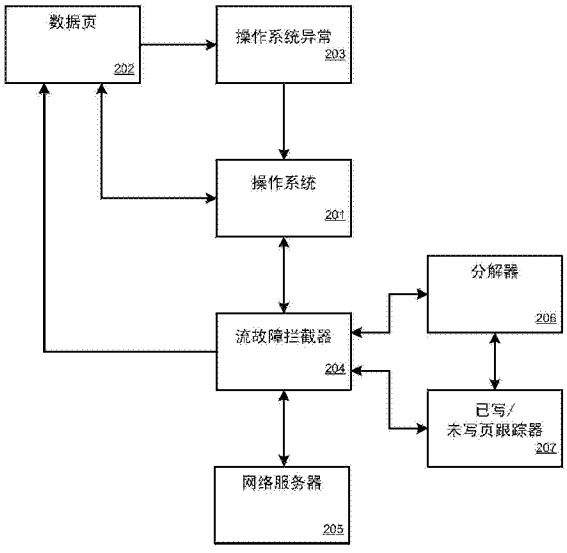
权利要求书1页 说明书6页 附图3页

(54)发明名称

可执行代码数据的无锁流传送的方法和系统

(57)摘要

分解器接收指令并将它们分解成多个分开的操作码。分解器创建标识每一个操作码之间的边界的表。通过原子地写存储器的标准块来将每一个操作码以逐操作码的方式写至存储器。当需要时将调试断点操作码追加到操作码以创建完整的存储器块。存储器块可以长达例如三十二位或者六十四位。长操作码可重叠两个或更多的存储器块。可将调试断点操作码追加到长操作码的第二部分以创建完整的存储器块。流故障拦截器标识何时所请求的数据页不可用并取回该数据页。



CN 104508626 B

1.一种计算机实现的方法,包括:

接收包括多个操作码的指令,每一个操作码包括一个或多个字节的数据;

将所述指令分解成分开的操作码;

创建标识每一个操作码之间的边界的表;

将调试断点操作码追加到操作码以创建完整的存储器标准块;以及

通过原子地写存储器的标准块将每一个操作码以逐操作码的方式写至存储器。

2.如权利要求1所述的计算机实现的方法,其特征在于,还包括:

将调试断点操作码写至所有存储器位置以初始化所述存储器。

3.如权利要求1所述的计算机实现的方法,其特征在于,所述存储器标准块长达三十二位或六十四位。

4.如权利要求1所述的计算机实现的方法,其特征在于,还包括:

标识何时长操作码将重叠两个或更多的存储器标准块;

将所述长操作码的第二部分写至第二存储器块;以及

将所述长操作码的第一部分写至第一存储器块。

5.如权利要求4所述的计算机实现的方法,其特征在于,还包括:

将调试断点操作码追加到所述长操作码的第二部分以创建完整的存储器标准块。

6.如权利要求5所述的计算机实现的方法,其特征在于,还包括:

将之前的操作码和所述长操作码的第一部分写至第一存储器块。

7.一种计算机系统,包括:

处理器;

系统存储器;

其上存储有计算机可执行指令的一个或多个计算机可读存储介质,所述计算机可执行指令在由所述一个或多个处理器执行时使得所述处理器执行一种用于将可执行代码流传送的方法,所述处理器操作以:

接收包括多个操作码的指令,每一个操作码包括一个或多个字节的数据;

将所述指令分解成分开的操作码;

创建标识每一个操作码之间的边界的表;

通过原子地写存储器的标准块将每一个操作码以逐操作码的方式写至存储器;以及

将调试断点操作码追加到操作码以创建完整的存储器标准块。

8.如权利要求7所述的计算机系统,其特征在于,所述处理器还操作以:

将调试断点操作码写至所有存储器位置以初始化所述存储器。

9.如权利要求7所述的计算机系统,其特征在于,所述处理器还操作以:

标识何时长操作码将重叠两个或更多的存储器标准块;

将所述长操作码的第二部分写至第二存储器块;以及

将所述长操作码的第一部分写至第一存储器块。

可执行代码数据的无锁流传送的方法和系统

技术领域

[0001] 本发明涉及一种计算机系统和计算机实现的方法，尤其涉及可执行代码数据的无锁流传送的方法和系统。

背景技术

[0002] 当同时收到读和写数据的命令时发生竞争状况。这也可以在例如在写线程将数据递送至存储器的同时一线程正在执行来自存储器的可执行数据分组的时候发生。结果可能是计算机崩溃、程序关闭、读/写错误或其他问题。在一个解决方案中，在等待写线程提供数据的同时可将执行线程挂起。然而，挂起线程可导致相关进程的性能问题。举例而言，对于进程可以有数十或数百个线程在运行，其中只有一个或几个线程访问该数据，但每一次流故障发生时所有的线程都必须被挂起以等待数据递送。

发明内容

[0003] 提供本概述以便以简化的形式介绍将在以下详细描述中进一步描述的一些概念。本概述并不旨在标识所要求保护主题的关键特征或必要特征，也不旨在用于限制所要求保护主题的范围。

[0004] 各实施例通过分解可执行代码并对于对存储器页的写排序来允许可执行代码以无锁的方式递送。这允许可执行代码贯穿流数据的递送而保持在一致的状态中。因为存储器处于一致的状态，所以当字节代码被写进存储器时其他线程可继续执行。这允许更大的可扩展性和性能。

[0005] 进一步的实施例启用用户模式中可执行代码数据的流传送，而无需挂起整个进程或使用内核模式组件。

附图说明

[0006] 为了进一步阐明本发明的各实施例的以上及其他优点和特征，将参考附图来呈现对本发明的各实施例的更具体的描述。应当理解，这些附图只描绘本发明的典型实施例，并且因此将不被认为是对其范围的限制。本发明将通过使用附图用附加特征和细节来描述和解释，在附图中：

[0007] 图1A-D示出了根据一个实施例的存储器写随时间的排序。

[0008] 图2示出了根据一个实施例用于流传送无锁可执行代码数据的系统。

[0009] 图3是根据一个实施例用于流传送可执行代码数据的过程或方法的流程图。

[0010] 图4示出了提供可执行代码数据的无锁流传送的适合的计算和网络环境的示例。

具体实施方式

[0011] 此处描述的无锁解决方案利用现代处理器的两个特征。首先，处理器提供原子三十二位或六十四位的存储器访问写。这保证了处理器将在存储器中看到所有的三十二位或

六十四位。第二，处理器有一个字节的调试断点异常操作码(opcode)。此调试断点操作码允许处理器重试从存储器读和执行操作码而不导致调用线程失败。

[0012] 在一个实施例中，在指令对处理器可用之前在存储器中分解代码。此分解允许创建包括数个单独的三十二位或六十四位写的存储器写的有序列表。此分解和分析可以在客户端或者服务器上执行。在客户端上执行此操作以增加处理器周期的代价节省网络传输，而在服务器上执行此操作意味着更大的网络传输但要求客户端节点处的较少处理器操作。

[0013] 当存储器页被分配时，整个页用一个字节的异常代码来写。流故障处理器保持已经写过或者无效的页和位置的列表。

[0014] 操作码使用从页末开始的分解的三十二位或六十四位的块写至存储器。替换性地，可以分析汇编树以标识叶操作码，叶操作码是由其他操作码调用的操作码。叶操作码可通过遍历操作码执行调用图来标识。这些叶操作码可首先被写至存储器。

[0015] 对于三十二位或更小(或者在其他实施例中是六十四位或更小)的操作码，整个操作码被写至存储器使得处理器将成功地执行指令。如果操作码大于三十二位(或者在其他实施例中是大于六十四位)，那么首先写操作码的“末尾”。通过首先写操作码的末尾，如果处理器在操作码仍在被写的时候执行指令，那么处理器将首先执行调试断点操作码。这允许流故障处理器完成写操作码并使得处理器可以重试该操作码。如果三十二位(或者六十四位)写跨多于一个操作码，那么调试断点操作码被编写成允许重试。如果三十二位(或者六十四位)写已经被写了，那么写入之前的值，在这种情况下操作码保持有效。

[0016] 通过写入由调试断点分隔的经分解的操作码，处理器将或者遇到无效的调试断点或者遇到有效的可执行代码。若应用程序执行无效的调试断点，那么如此处所述的流故障拦截器将在异常被递送到应用程序之前拦截该异常。流故障拦截器等待直到存储器有效，然后重试操作码。流故障拦截器可以使用页和被写的操作码的表来区分作为流故障的一部分而发生、必须被重试的断点或者由调试器或应用程序使用、必须被传回应用程序的断点。在有效可执行代码的情况中，应用程序成功地执行代码。结果是，流故障处理器不需要在递送流故障的结果时挂起进程中的所有线程。

[0017] 图1A-D示出了根据一个实施例的存储器写随时间的排序。在此示例中使用的程序代码包括指令：0xAB 0xE9 0x00 0xFE 0x70 0x08…。分解器将此代码拆分成以下操作码：

[0018] (1)：0xAB

[0019] (2)：0xE9 0x00 0xFE 0x70 0x08

[0020] (3)：....

[0021] 第一个操作码长达一个字节，第二个操作码长达五个字节，且不同长度的其它操作码是以类似的方式标识的。

[0022] 存储器101包括数个三十二位的块102-104。在其他实施例中，存储器块可以是六十四位或某一其他大小。图1A示出了存储器101全部用一个字节的调试断点操作码0xCC填充的原始状态。如果线程在此时从存储器101读取指令，则调试断点操作码将导致流处理器试图将代码加载到存储器101于是线程将重试读取代码。

[0023] 在图1B中，第一个操作码0xAB已经被写至三十二位的块102b。第一个、一个字节长的操作码0xAB用替代第一个0xCC的原子写来写入到存储器101中。写操作需要写三十二位的数据。因为指令在写入存储器之前已经被分解了，写操作跟踪正在写什么字节并知道用

0xCC填充块102b剩下的二十四个位以填满整个三十二位的块。如果处理器执行第一个指令0xAB，它将会成功。但是，如果处理器试图在此时执行第二个指令，则调试断点0xCC将被读取并且流故障重试机制将接管。

[0024] 现在流故障处理器需要写第二个操作码(即，0xE9 0x00 0xFE 0X70 0x08)。流故障处理器知道前一指令(0xAB)被写了并且知道块102b-104b是怎样被写的。流故障处理器也知道下一操作码是多长和它需要怎样被写至存储器101。特别是，流故障处理器知道第二个、五个字节长的操作码将在操作码0xAB之后被写至块102b和103b。流故障处理器也判断第二个操作码将不能放进单个三十二位块102b的剩余空间中。因而，第二个操作码必须在存储器块102b和103b之间拆分。

[0025] 如图1C中所示，在写操作码的第一部分之前，操作码的第二部分(即，0x70 0x08)以调试断点0x00补充以填满三十二位而写至块103c。通过首先写操作码的第二部分，如果处理器执行0xAB块之后的操作码，那么处理器将招致调试故障(0xCC)并将不会试图执行垃圾。

[0026] 在操作码的第二部分被写至块103c之后，然后第一部分(即，0xE9 0x000xFE)可以被写至存储器101。如图1D中所示，此第一部分随第一操作码0xAB写入它之前在块102d中所分配的位置。此时，处理器将能够成功地执行两个指令。将指令写到存储器101的这个过程将被逐操作码地应用直到整个页面常驻。

[0027] 图2示出了根据一个实施例用于流传送无锁可执行代码数据的系统。在操作系统201上运行的应用程序试图从存储器读数据页202。当调用了可执行存储器的非常驻页时，这触发操作系统异常203，该异常通知操作系统201说这个调用失败了。在应用程序看到故障之前，操作系统201向流故障拦截器204发送读失败返回码。流故障拦截器204在它的数据库中查找该页并判断它需要取回该页。流故障拦截器204进行网络调用以从例如网络服务器205得到页面，并改变该页的存储器访问使得它可以将数据写至数据页202。

[0028] 在已有的系统中，流故障拦截器204将需要挂起请求数据的进程。如果进程没有被挂起，那么另一个线程可以如同页202上的数据被写入存储器一样地执行该数据，并且当进程试图执行部分数据时将崩溃。在页202被写至存储器之后，进程可以恢复，流故障拦截器将操作系统异常改为“成功”且应用程序继续执行。

[0029] 在已有的系统中，挂起整个进程的这个步骤是必要的，因为改变对数据页202的访问并将数据写至数据页202不能在单个步骤中完成。这意味着当服务于流故障时，进程中的两个线程不可能在执行。

[0030] 本文通过添加分解器206和已写/未写页跟踪器207改进了已有的系统。分解器206负责取得返回的字节代码并分解它。分解器206返回包含每一个操作码的边界的偏移列表。如图1的示例中所注意到的，每一个操作可以长一个或多个字节。已写/未写页跟踪器207包含由分解器206生成的偏移列表和已经提交了的那些偏移的列表。

[0031] 当使用分解器206和已写/未写页跟踪器207的时候进程不需要被挂起。相反，在将所请求的代码从服务器205返回给流故障拦截器204之后，分解器被调用以生成偏移地图。然后偏移地图由流故障拦截器204存储，该流故障拦截器开始以如上所述的逐操作码的方式将操作码写至数据页202。流故障拦截器204咨询此偏移地图以确定每一个操作码的边界。流故障拦截器204按照任何合适的从操作码列表的开头或末尾开始的排序将操作码写

至数据页202。

[0032] 如果发生任何争用诸如处理器执行部分填充的指令时，那么操作系统故障机制203将被触发。这将导致流故障拦截器204被操作系统201调用。流故障拦截器204咨询已写/未写页跟踪器207并或者将操作码本身写至数据页202或者等待直到另一个线程完成了写数据。然后流故障拦截器204重试异常。

[0033] 图3是根据一个实施例用于流传送可执行代码数据的过程或方法的流程图。在步骤301，处理器接收包括多个操作码的指令。每一个操作码都可包括一个或多个字节的数据。在步骤302，使用分解器以将指令分解成分开的操作码。在步骤303，创建标识每一个操作码之间的边界的表。表可存储在例如已写/未写页跟踪器中。在步骤304，通过原子地写存储器的标准块来将每一个操作码以逐操作码的方式写至存储器。操作码可由例如流故障拦截器写至存储器。

[0034] 在一些实施例中，可将调试断点操作码写至所有存储器位置以初始化存储器。当操作码被写至存储器时，可将调试断点操作码追加到操作码以创建完整的存储器标准块。存储器标准块可以长达例如三十二位或者六十四位。

[0035] 当长操作码将与两个或多个存储器标准块重叠时，可将它分成两个部分。将长操作码的第二部分写至第二存储器块，然后将长操作码的第一部分写至第一存储器块。以这种方式写长操作码将保证，如果在写完整个操作码之前读存储器位置，将遇到调试断点。可将调试断点操作码追加到长操作码的第二部分以创建完整的存储器标准块。可将之前的操作码与长操作码的第一部分一起写至第一存储器块。

[0036] 应理解，图3中所示的过程的步骤301-305可同时地和/或顺序地执行。还应理解，每一个步骤都可以任何次序执行且可被执行一次或重复地执行。

[0037] 图4输出了适合的计算和网络环境400的示例，在其上可实现图1-3的示例以提供可执行代码数据的无锁流传送。计算系统环境400仅是适合的计算环境的一个示例，并不旨在建议对本发明的使用或功能性的范围的任何限制。本发明是可以多种其他通用或专门计算系统环境或配置操作。适合在本发明中使用的公知的计算系统、环境和/或配置的示例包括，但不限于，个人计算机、服务器计算机、手持或膝上型设备、平板设备、多处理器系统、基于微处理器的系统、机顶盒、可编程消费者电子产品、网络PC、小型机、大型机、包含上述系统或设备中的任一个的分布式计算机环境等。

[0038] 本发明可在诸如程序模块等由计算机执行的计算机可执行指令的通用上下文中描述。一般而言，程序模块包括执行特定任务或实现特定抽象数据类型的例程、程序、对象、组件、数据结构等等。本发明也可以在其中任务由通过通信网络链接的远程处理设备执行的分布式计算环境中实现。在分布式计算环境中，程序模块可以位于包括存储器存储设备在内的本地和/或远程计算机存储介质中。

[0039] 参考图4，用于实现本发明的各方面的示例性系统可包括计算机400形式的通用计算设备。组件可包括但不限于，各种硬件组件诸如处理单元401、数据存储402诸如系统存储器、和将包括数据存储402的各种系统组件耦合到处理单元401的系统总线403。系统总线403可以是若干类型的总线结构中的任一种，包括使用各种总线体系结构中的任一种的存储器总线或存储器控制器、外围总线、以及局部总线。作为示例而非限制，这样的体系结构包括工业标准体系结构(ISA)总线、微通道体系结构(MCA)总线、增强型ISA(EISA)总线、视

频电子标准协会(VESA)局部总线,以及也称为夹层(Mezzanine)总线的外围部件互连(PCI)总线。

[0040] 计算机400通常包括各种计算机可读介质404。计算机可读介质404可以是能由计算机400访问的任何可用介质,并包含易失性和非易失性介质以及可移动、不可移动介质,但不包括所传播的信号。作为示例而非限制,计算机可读介质404可包括计算机存储介质和通信介质。计算机存储介质包括以存储诸如计算机可读的指令、数据结构、程序模块或其他数据之类的信息的任何方法或技术实现的易失性和非易失性、可移动和不可移动介质。计算机存储介质包括,但不仅限于,RAM、ROM、EEPROM、闪存或其他存储器技术、CD-ROM、数字多功能盘(DVD)或其他光盘存储、磁带盒、磁带、磁盘存储或其他磁存储设备,或可以用来存储所需信息并可以被计算机400访问的任何其他介质。通信介质通常以诸如载波或其他传输机构之类的已调制数据信号来具体化计算机可读指令、数据结构、程序模块或其他数据,并且包括任何信息传送介质。术语"已调制数据信号"是指具有以在信号中编码信息的方式被设定或改变其一个或多个特征的信号。作为示例而非限制,通信介质包括诸如有线网络或直接线连接之类的有线介质,以及诸如声学、RF、红外及其他无线介质之类的无线介质。上面各项中的任何项的组合也包括在计算机可读介质的范围内。计算机可读介质可体现为计算机程序产品,诸如存储在计算机存储介质上的软件。

[0041] 数据存储或系统存储器402包括易失性和/或非易失性存储器形式的计算机存储介质,诸如只读存储器(ROM)和随机存取存储器(RAM)。数据存储402或计算机可读介质404可用来存储数据页、操作码边界列表、操作码等。基本输入/输出系统(BIOS)通常存储在ROM中,包含了诸如在启动过程中帮助在计算机400内的元件之间传输信息的基本例程。RAM通常包含处理单元401可立即访问和/或当前正在操作的数据和/或程序模块。作为示例而非限制性,数据存储402保存操作系统、应用程序、及其他程序模块和程序数据。运行在处理单元401上的操作系统可支持诸如操作系统异常203、流故障拦截器204、分解器206和/或已写/未写页跟踪器207的功能(图2)。

[0042] 数据存储402也可包括其他可移动/不可移动、易失性/非易失性计算机存储介质。仅作为示例,数据存储402可以是对不可移动、非易失性磁介质进行读写的硬盘驱动器、对可移动、非易失性磁盘进行读写的磁盘驱动器、以及对诸如CD ROM或其它光学介质等可移动、非易失性光盘进行读写的光盘驱动器。可在示例性操作环境中使用的其他可移动/不可移动、易失性/非易失性计算机存储介质包括但不限于,磁带盒、闪存卡、数字多功能盘、数字录像带、固态RAM、固态ROM等。以上描述并在图4中示出的驱动器及其相关联的计算机存储介质为计算机400提供了对计算机可读指令、数据结构、程序模块和其他数据的存储。

[0043] 用户可通过诸如平板、电子数字化仪、话筒、键盘和/或定点设备(通常指的是鼠标、跟踪球或触摸垫)等用户接口405或其他输入设备输入命令和信息。其他输入设备可包括操纵杆、游戏手柄、圆盘式卫星天线、扫描仪等等。附加地,语音输入、使用手或手指的姿势输入、或其他自然用户接口(NUI)页可与合适的输入设备诸如话筒、相机、平板、触摸板、手套或其他传感器一起使用。这些及其他输入设备常常通过耦合到系统总线403的用户接口405连接至处理单元401,但也可通过其他接口和总线结构,诸如并行端口、游戏端口或通用串行总线(USB)端口来进行连接。监视器406或其他类型的显示设备也经由诸如视频接口的接口连接至系统总线403。监视器406也可以与触摸屏面板等集成。注意到监视器和/或触

摸屏面板可以在物理上耦合至其中包括计算设备400的外壳，诸如在平板型个人计算机中。此外，诸如计算设备400的计算机页可包括其他外围输出设备，诸如扬声器和打印机，它们可以通过输出外围接口等连接。

[0044] 计算机400可使用至一个或多个诸如远程计算机的远程设备的网络接口407在网络化或云计算的环境中操作。远程计算机可以是个人计算机、服务器、路由器、网络PC、对等设备或其它常见的网络节点，并且通常包括上面相对于计算机400所述的许多或全部元件。图4中所示的网络接口包括一个或多个局域网(LAN)和一个或多个广域网(WAN)，但也可以包括其他网络。此类联网环境在办公室、企业范围的计算机网络、内联网和因特网中是常见的。

[0045] 当用于网络化或云计算的环境中时，计算机400可通过网络接口或适配器407连接至公共或私有网络。网络接口407可提供至诸如网络服务器205(图2)的远程设备的连接。在一些实施例中，用于通过网络建立通信的调制解调器或其他装置。调制解调器可以是内置的或外置的，可经由网络接口407或其他适当的机制连接至系统总线403。诸如包括接口和天线的无线联网组件可通过诸如接入点或对等计算机等合适的设备耦合到网络。在联网环境中，相对于计算机400所示的程序模块或其部分可被存储在远程存储器存储设备中。可以理解，所示的网络连接是示例性的，也可以使用在计算机之间建立通信链路的其他手段。

[0046] 尽管用结构特征和/或方法动作专用的语言描述了本主题，但可以理解，所附权利要求书中定义的主题不必限于上述具体特征或动作。更确切而言，上述具体特征和动作是作为实现权利要求的示例形式公开的。

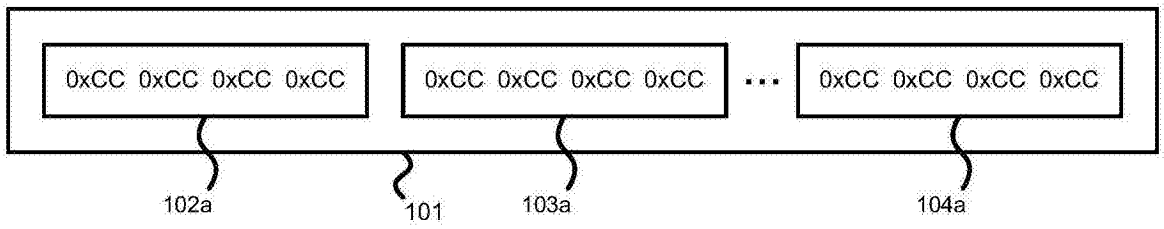


图1A

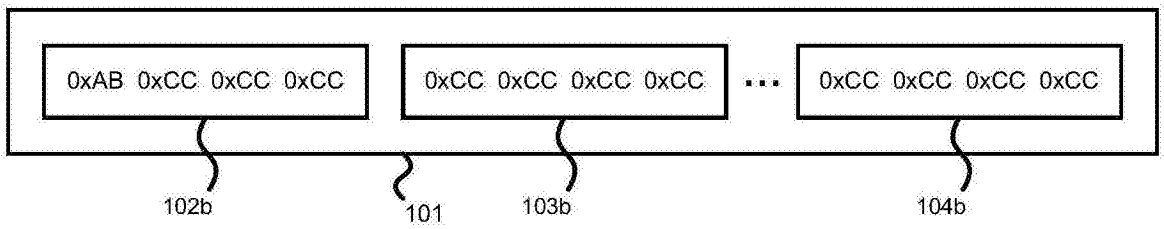


图1B

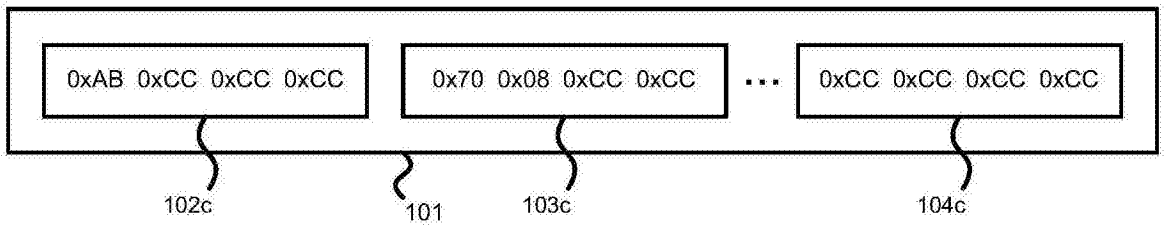


图1C

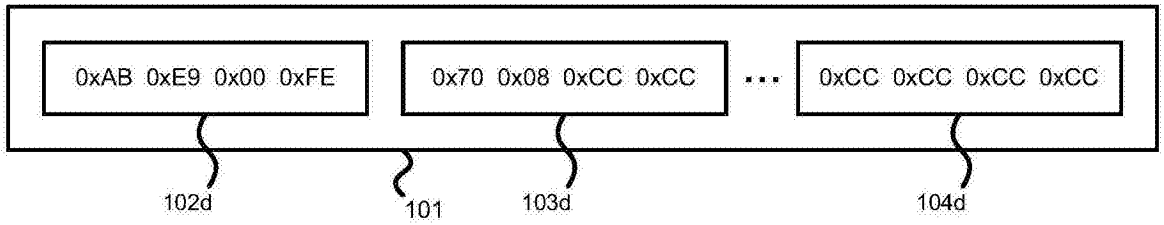


图1D

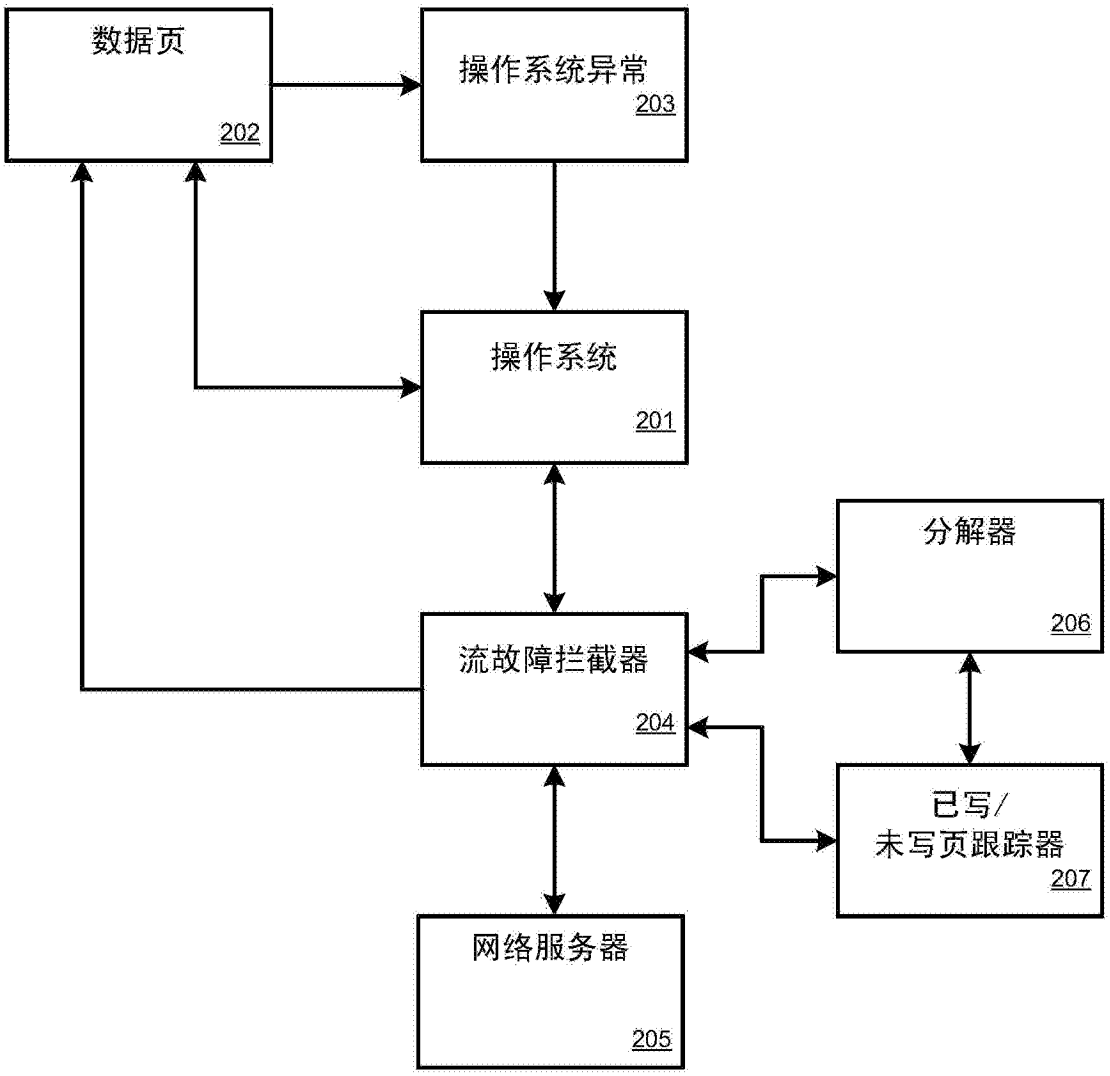

数据页
202
操作系统异常
203
操作系统
201
分解器
206
流故障拦截器
204
已写/
未写页跟踪器
207
网络服务器
205

图2

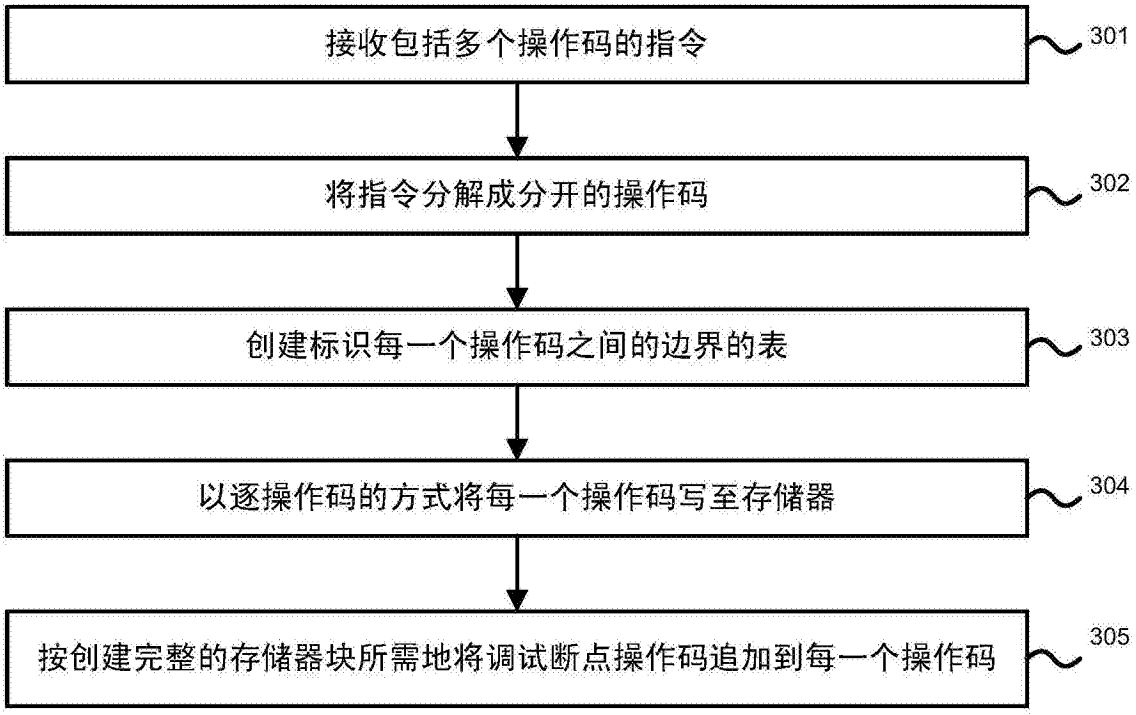

接收包括多个操作码的指令
301
将指令分解成分开的操作码
302
创建标识每一个操作码之间的边界的表
303
以逐操作码的方式将每一个操作码写至存储器
304
按创建完整的存储器块所需地将调试断点操作码追加到每一个操作码
305

图3

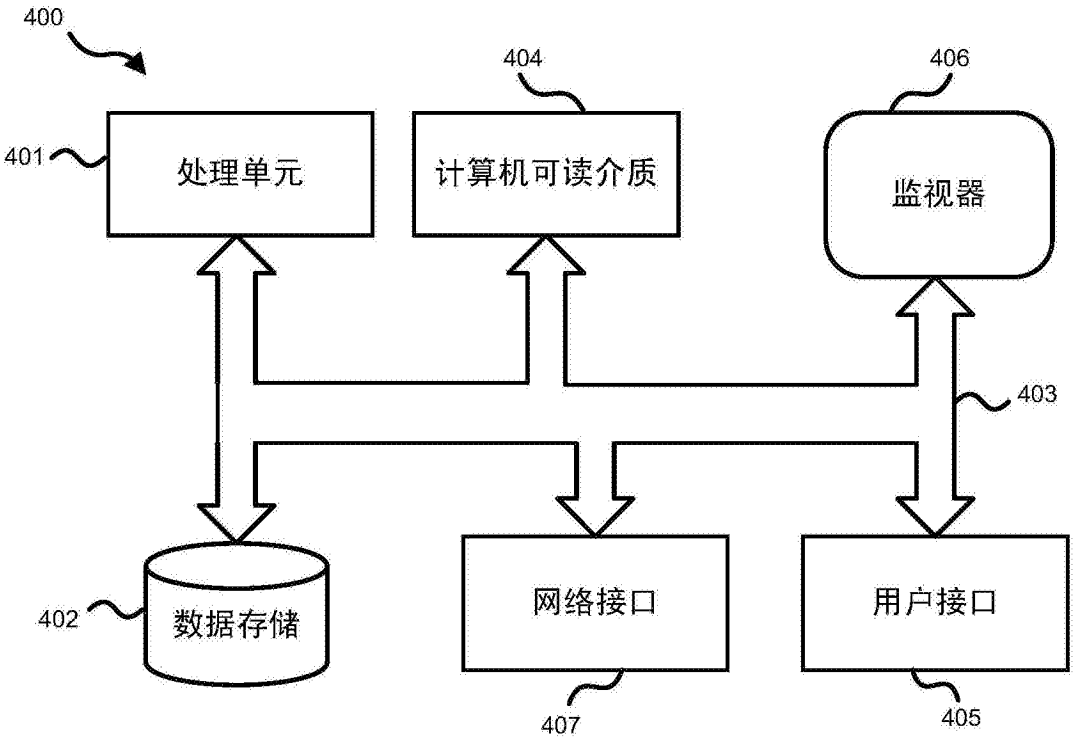

400
401
处理单元
404
计算机可读介质
406
监视器
403
402
数据存储
网络接口
407
用户接口
405

图4