



(12)发明专利

(10)授权公告号 CN 104205049 B

(45)授权公告日 2018.05.11

(21)申请号 201280071658.0

(74)专利代理机构 中国专利代理(香港)有限公司 72001

(22)申请日 2012.03.22

代理人 徐予红 汤春龙

(65)同一申请的已公布的文献号

申请公布号 CN 104205049 A

(51)Int.Cl.

G06F 8/41(2018.01)

(43)申请公布日 2014.12.10

G06F 9/445(2018.01)

(85)PCT国际申请进入国家阶段日

2014.09.22

(56)对比文件

US 2009/0094586 A1, 2009.04.09,

(86)PCT国际申请的申请数据

US 2009/0094586 A1, 2009.04.09,

PCT/CN2012/072780 2012.03.22

US 7478373 B2, 2009.01.13,

(87)PCT国际申请的公布数据

US 2002/0013892 A1, 2002.01.31,

W02013/139015 EN 2013.09.26

US 2002/0052727 A1, 2002.05.02,

(73)专利权人 英特尔公司

US 2007/0011666 A1, 2007.01.11,

地址 美国加利福尼亚州

审查员 谢萍

(72)发明人 X.林 Y.金 Y.吴 J.李 L.林

权利要求书2页 说明书9页 附图5页

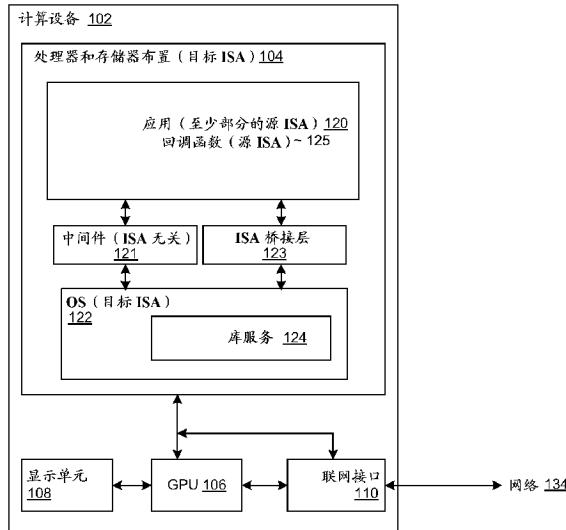
X 施

(54)发明名称

混合模拟和内核函数处理系统和方法

(57)摘要

一个实施例预先构建内核函数(KF)的解释并将其加载到解释池和对应索引表中。由此将KF快速地加载，并且不一定要等待通过LIB模拟器来诱捕和模拟。这促成更快速地访问KF。其他实施例提供混合模拟，其中一些应用函数(例如，需要快速性能的那些应用函数)从源ISA库解释，而其他应用函数通过对目标ISA库的模拟来处理。这样提供对某些函数的快速访问。本文描述了其他实施例。



1. 一种由至少一个处理器执行的方法,其包括:
加载具有源指令集体系结构ISA的应用;
将第一源指令的第一解释存储在至少一个存储器,所述第一解释是从所述源ISA到目标ISA;
在存储所述第一解释之后,(a)检索存储的第一解释并执行所述解释,以及(b)未成功尝试找到第二源指令的第二解释,以及此后,确定所述第二解释并执行所述第二解释。
2. 如权利要求1所述的方法,其中所述第一源指令包括内核函数KF。
3. 如权利要求2所述的方法,其中存储所述第一解释包括以具有用于所述至少一个存储器中存储的解释的索引的格式来存储所述第一解释。
4. 如权利要求2所述的方法,其中将所存储的第一解释固定以避免从所述至少一个存储器移除所述第一解释。
5. 如权利要求2所述的方法,其中所述第二解释在所述应用的运行时期间来确定。
6. 如权利要求2所述的方法,其中从不模拟所述第一源指令以及为所述目标ISA模拟所述第二源指令。
7. 如权利要求1所述的方法,其包括响应于用户选择来指定用于解释的第一未定义符号以及指定用于模拟的第二未定义符号。
8. 如权利要求1所述的方法,其包括解释而不是模拟第一未定义符号以及模拟第二未定义符号。
9. 如权利要求8所述的方法,其中所述第一和第二未定义符号分别地对应于第一和第二匿名全局变量。
10. 如权利要求8所述的方法,其包括通过指向目标ISA库的门来模拟第二未定义符号。
11. 如权利要求8所述的方法,其包括与源ISA库共享目标ISA库中存储的命名全局。
12. 如权利要求1所述的方法,其包括通过解释而不是模拟来解析第一未定义符号,以及通过模拟来解析第二未定义符号。
13. 一种计算装置,其包括:
用于加载具有源指令集体系结构ISA的应用的模块;
用于将第一源指令的第一解释存储在至少一个存储器的模块,所述第一解释是从所述源ISA到目标ISA;
用于在存储所述第一解释之后,(a)检索存储的第一解释并执行所述解释,以及(b)未成功尝试找到第二源指令的第二解释,以及此后,确定所述第二解释并执行所述第二解释的模块。
14. 如权利要求13所述的计算装置,其中所述第一源指令包括内核函数KF。
15. 如权利要求14所述的计算装置,其中用于存储所述第一解释的模块包括用于以具有用于所述至少一个存储器中存储的解释的索引的格式来存储所述第一解释的模块。
16. 如权利要求14所述的计算装置,其中将所存储的第一解释固定以避免从所述至少一个存储器移除所述第一解释。
17. 如权利要求14所述的计算装置,其中所述第二解释在所述应用的运行时期间来确定。
18. 如权利要求14所述的计算装置,其中从不模拟所述第一源指令以及为所述目标ISA

模拟所述第二源指令。

19. 如权利要求13所述的计算装置,其包括用于响应于用户选择来指定用于解释的第一未定义符号以及指定用于模拟的第二未定义符号的模块。

20. 如权利要求13所述的计算装置,其包括用于解释第一未定义符号而不模拟以及模拟第二未定义符号的模块。

21. 如权利要求20所述的计算装置,其中所述第一和第二未定义符号分别地对应于第一和第二匿名全局变量。

22. 如权利要求20所述的计算装置,其包括用于通过指向目标ISA库的门来模拟第二未定义符号的模块。

23. 如权利要求20所述的计算装置,其包括用于与源ISA库共享目标ISA库中存储的命名全局的模块。

24. 如权利要求13所述的计算装置,其包括用于通过解释而不是模拟来解析第一未定义符号以及通过模拟来解析第二未定义符号的模块。

25. 一种通信装置,包括执行如权利要求1-12中任一项所述的方法的部件。

26. 一种计算设备,其包括:用于执行如权利要求1至12中任一项的部件。

27. 一种计算设备,其包括:

应用,所述应用具有存储在至少一个存储器中的源指令集体体系结构ISA;

至少一个处理器,所述至少一个处理器具有目标ISA且耦合到所述至少一个存储器,用于执行包括如下的操作:

加载所述应用;

将第一源指令的第一解释存储在所述至少一个存储器,所述第一解释是从所述源ISA到目标ISA;

在存储所述第一解释之后,(a)检索存储的第一解释并执行所述解释,以及(b)未成功尝试找到第二源指令的第二解释,以及此后,确定所述第二解释并执行所述第二解释。

28. 如权利要求27所述的计算设备,其中所述第一源指令包括内核函数KF。

29. 如权利要求28所述的计算设备,其中所述第二解释在所述应用的运行时期间来确定。

30. 如权利要求28所述的计算设备,其中所述操作包括解释第一未定义符号而不模拟以及模拟第二未定义符号。

31. 一种包括如权利要求27至30中任一项所述的至少一个处理器的计算机系统,所述计算机系统还包括显示器。

32. 至少一种机器可读介质,其包括:多个指令,响应所述多个指令在计算设备上被执行,促使所述计算设备执行如权利要求1至12中任一项所述的方法。

混合模拟和内核函数处理系统和方法

背景技术

[0001] 计算设备可以由其指令集体系结构(ISA)来表征。典型地,计算设备可以包括操作系统(OS)服务,以及OS服务可以包括针对计算设备的ISA开发的运行时库服务(LIB),以便帮助应用开发者要在该计算设备上运行的应用。如果应用是针对该计算设备的ISA以外的ISA编写的,则需要对该应用模拟。确切的来说,模拟使(针对第一ISA编写的)应用在计算设备的(使用第二ISA的)体系结构上执行并访问目标平台的运行时LIB服务。再者,应用的ISA从属部分可以包括回调函数(例如,从ISA从属运行时LIB返回到被模拟应用进行调用的函数、返回到需要被模拟的源LIB服务的函数)。

[0002] 但是,模拟可能由于ISA之间没有硬件能力而遇到难题。例如,源ISA可以实现底层目标ISA未提供的某种功能性。此类功能性可以包括原子操作,原子操作是不可见且不可约的,以致于原子操作必须整体地执行或完全不执行(例如,此类操作可以包括处理器在同一个总线操作中同时读取位置和向位置写入)。

[0003] 在其他情况下,可以采用如内核函数(KF)的项目,其独立于源或目标ISA。KF不是ISA提供的功能性,而是与底层平台的OS提供的功能性相关(例如,下文进一步论述的LIB服务124)。KF的模拟可能不会由于如上文论述的没有硬件能力而是由于源和目标LIB服务之间的不同能力而遇到难题。KF可能包含访问内核数据并在内核空间中运行但是不需要源应用与其主机模拟机之间的上下文切换的函数。由此,使用KF能够快速地访问内核数据,而且还免去访客与主机模拟机或环境之间的上下文切换的开销。然而,如果主机模拟机未正确地捕获KF,则KF仍可能导致上下文切换的开销。

[0004] 模拟还可能由于要模拟的内容的大小而遇到难题。例如,移动平台可能面对在仅选择正真需要模拟的功能时模拟整个库(可能非常大)的难题。

附图说明

[0005] 将通过附图中图示的示范实施例而非限制来描述本发明的实施例,在附图中相似的引用表示相似的元件,以及其中:

[0006] 图1图示本发明实施例中包括ISA桥接的计算设备;

[0007] 图2进一步详细地图示图1的ISA桥接层;

[0008] 图3和图4图示本发明实施例中用于桥接源ISA的应用与目标ISA的LIB服务之间的调用和回调的方法;

[0009] 图5包括用于高效KF处理的实施例;

[0010] 图6-7包括混合模拟的实施例。

具体实施方式

[0011] 多种操作将描述为多个离散操作,进而以最有助于理解这些说明性实施例的方式来描述这些多种操作;但是,描述的次序不应视为暗示这些操作必定是按照次序的。具体来说,这些操作无需按出现的次序来执行。再者,将多个操作描述为单独的操作不应视为要求

这些操作必定要独立地被执行和/或由单独的实体来执行。将多个实体和/或模块描述为单独的模块同样地不应视为要求这些模块是单独的和/或执行单独的操作。在多种实施例中，可以将图示和/或描述的操作、实体、数据和/或模块合并、将其进一步拆分成从属部件和/或将其省略。短语“实施例”被反复地使用。该短语一般不是指同一个实施例；但是，它可以指同一个实施例。除上下文另行指示，否则术语“包括”、“具有”和“包含”是同义词。短语“A/B”表示“A或B”。短语“A和/或B”表示“(A)、(B)或(A和B)”。短语“A、B和C的至少其中之一”表示“(A)、(B)、(C)、(A和B)、(A和C)、(B和C)或(A、B和C)”。

[0012] 一个实施例预先构建KF的解释并将其加载到解释池和对应索引表中。由此将KF快速地加载，并且不一定要等待通过LIB模拟器来诱捕和模拟。这促成更快速地访问KF。其他实施例提供混合模拟，其中一些应用函数（例如，需要快速性能的那些应用函数）从源ISA库解释，而其他应用函数通过对目标ISA库的模拟来处理。这样提供对某些函数的快速访问。

[0013] 图1图示实施例中包括ISA桥接（可选地具有回调）的示例计算设备。计算设备102可以包括处理器和存储器布置104，处理器和存储器布置104包括或耦合到OS 122、ISA桥接层123、应用120、图形处理单元（GPU）106、显示单元108和联网接口110，它们如图所示地彼此耦合（即，直接地或间接地耦合）。OS 122可以包括服务124的LIB。计算设备102还可以在应用120与OS 122之间包括可选中间件121。正如下文将更详细描述的，ISA桥接层123可以配置有多种运行时特征和服务以使得应用120能够整体地或部分地（例如，当还使用ISA无关的中间件121时）在源ISA中实现，而OS 122（包括LIB服务124）可以在与源ISA不同的目标ISA中实现。再者，应用120可以是包括多种情况下需要LIB服务124中的多种库服务以“回调”应用120的多种回调函数125的库服务124的应用（具体来说，使用源ISA实现的部分）。ISA桥接层123也可以在本文中称为进程虚拟机（PVM）。

[0014] 计算设备102可以是服务器、桌上型计算机、膝上型计算机、平板计算机、智能电话、个人数字助理、游戏控制台、因特网电器、移动上网设备、蜂窝电话、移动联网设备、移动计算节点或其他接设备。处理器和存储器布置104表示宽范围的处理器和存储器布置，其包括具有多种执行速度和功耗的单核或多核处理器和多种体系结构（例如，具有一级或多级高速缓存）以及多种类型的存储器的布置。在多种实施例中，GPU 106可以配置成为OS 122提供视频解码和/或图形处理功能，而显示单元108可以配置成能够在其上呈现多媒体内容，（例如HD视频）。相似地，GPU 106和显示单元108旨在表示本领域中公知的宽范围的图形处理器和显示元件。同样地，网络134旨在表示宽范围的网络。网络134的示例可以包括有线网络或无线网络、局域网或广域网、专用网络或公用网络，包括因特网。OS 122（包括LIB服务124），定义LIB服务124的调用的应用编程接口（API）除外，表示本领域中公知的宽范围的OS元件。OS 122可以包括常规组件，如配置成管理存储器资源、调度任务执行等的内核以及配置成管理多种设备资源的设备驱动器。在多个实施例中，OS 122可以包括支持可选中间件121的虚拟机（例如安卓™应用框架支持的安卓™虚拟机）。除了定义LIB服务124的调用外，为了帮助调用应用120的回调函数125，LIB服务124的API还可以包括应用120的回调函数125的对应存根（stub）和签名。OS 122的示例可以包括Windows®操作系统、Linux®、安卓™、IOS®等。相似地，可选中间件121旨在表示宽范围的中间件元件，包括但不限于ISA无关的中间件。中间件121的示例可以包括但不限于，安卓™应用框架、Java™或其他应用框架或ISA无关的执行环境。同样地，应用120（包括回调功能125）旨在表示宽范围的应用，包括

用于个人助理、生产力、社交网络应用、日程表、字处理、电子表格、Twitter®、Facebook®、浏览器等。

[0015] 在图2中，ISA桥接层123可以包括ISA桥接加载器202、源ISA模拟器204和目标ISA LIB模拟器206(其配置成提供多种运行时特征和服务，包括但不限于动态绑定服务)。源ISA模拟器204可以包括源ISA上下文212和二进制解释引擎215。源ISA模拟器204可以保持在源ISA上下文212中，源ISA体系结构的执行上下文包括但不限于例如，当前执行指令指针(IP)。二进制解释器引擎215可以配置成将源ISA指令解释为目标ISA指令。LIB模拟器206可以包括目标ISA LIB上下文222、门224(例如，处理器用于控制对特权函数的访问、更改数据段、切换表等的数据结构)以及封装器函数226。LIB模拟器206可以保持在目标ISA库(LIB)上下文222、目标ISA LIB 124的执行上下文中。在多种实施例中，对于每个LIB服务124(例如，函数)可以有对应的一对门224和封装器函数226，其中该对配置成帮助应用120跨源和目标ISA体系结构调用LIB服务124。相似地，每个回调函数125可以有对应的一对门224和封装器函数226，其配置成帮助LIB服务124跨源和目标ISA体系结构对回调函数125进行回调。

[0016] ISA桥接加载器202可以是配置成将应用120加载到存储器中的实用工具。在加载应用120时，ISA桥接加载器202可以配置成解析与源应用120所对应的LIB关联的应用120的任何未解析的符号126。符号可以是寄存器的标识符(例如，文本字符串)、存储器地址等。ISA桥接加载器202可以配置成将符号修改(到回调函数125)，并将回调函数125的符号关联到对应封装器函数226。ISA桥接加载器202可以采用多种公知方式的任何一种方式从OS 122(或从中间件121(如果采用的话))加载器获得加载的控制，这些公知的方式包括在OS 122或中间件121支持时，使用基于二进制格式的控制转移或加载/预加载变量。在其他实施例中，可以修改OS 122(或中间件121(如果采用的话))的加载器以帮助将控制转移到ISA桥接加载器202。

[0017] 源ISA模拟器204可以在目标ISA 122的“顶层”上模拟源ISA 120”以运行源ISA应用120。正如先前描述的，源ISA模拟器204可以配置成保持源ISA执行上下文212。例如，源ISA模拟器204可以配置成在应用120的执行期间跟踪源ISA IP。当应用120尝试调用LIB服务124时，源ISA模拟器204可以在监视源ISA执行，并且继而可以调用并转移执行控制到LIB模拟器206。在多种实施例中，源ISA模拟器204可以调用并转移执行控制到LIB服务124的对应门224(下文予以进一步论述)。

[0018] LIB模拟器206可以通过映射到目标LIB 124来模拟源LIB(或任何其他LIB)。再有，LIB模拟器206可以配置成保持目标ISA LIB(LIB)执行上下文222。与LIB服务124对应的门224可以配置成分别地将对LIB服务124的调用重定向到对应封装器函数226，这些对应封装器函数226处理和设置这些调用。而与回调函数125对应的门224可以配置成分别地将回调的执行控制从对应封装器函数226转移到源ISA模拟器204。在多种实施例中，每个门224可以包括配置成实施向对应封装器函数226或源ISA模拟器204重定向的指令。在多种实施例中，每个门224的指令可以是配置成与二进制解释引擎215协作以实施执行控制重定向的源ISA指令。在多种实施例中，每个门224还可以包括标识对应封装器函数226的指示符。

[0019] 在多种实施例中，为了处理和设置对对应的LIB服务124的调用，与LIB服务124对应的每个封装器函数226可以配置成从源ISA上下文212检索调用的关联的参数值，将调用从源ISA应用二进制接口(ABI)格式转换成目标ISA ABI格式，并将转换的调用附带参数值

保存在LIB上下文222中。在回调到回调函数125时,可以将执行控制转移到回调函数125的对应封装器函数226。在多种实施例中,为了处理和设置对应用120的回调函数125的回调,与回调函数125对应的每个封装器函数226可以配置成将回调从目标ISA ABI格式转换成源ISA ABI格式,连接回调的关联的参数值,并将转换的回调附带参数值保存在源ISA上下文212中。与回调函数125对应的门224可以配置成利用(封装器函数226准备且与回调函数125对应的)源ISA上下文212来调用源ISA模拟器204以便模拟源ISA格式在目标ISA 104中呈示的回调函数。

[0020] 图3-4图示根据本发明实施例的具有回调的示例ISA桥接方法。方法300可以包括两个部分,用于将调用从(源ISA的)应用120桥接到(目标ISA的)LIB服务124的部分300a,以及用于将回调从(目标ISA的)LIB服务124桥接到(源ISA的)应用120的回调函数125的部分300b。部分300a和300b可以彼此独立地来实现。再有,多种实施例无需关心封装器函数和/或回调函数,而是可以将重点放在本文描述的实施例的其他方面。

[0021] 在图3中,在框302处,ISA桥接加载器202可以加载源应用120。在加载应用120时,ISA桥接加载器202可以解析至LIB服务124的符号名称或引用,并修改回调函数125的符号名称或引用,正如早前描述。在框304处,在执行过程中,应用120可以调用LIB服务124的其中之一。在多种实施例中,应用120可能需要被调用的LIB服务124向其回调函数125的其中之一回调。在多个实施例中,应用120可以作为对被调用的LIB服务124的调用的一部分包括至回调函数125的指针。并不将该指针传递到回调函数125,而是LIB服务124的封装器函数226可以传递回调函数125的对应封装器函数226。在框306处,源ISA模拟器204在(例如通过监视源ISA IP并确定IP正在引用目标LIB的地址范围内的地址)检测到调用时,源ISA模拟器204可以将该调用重定向到LIB模拟器206中的LIB服务124的对应门224并将执行控制转移到LIB模拟器206中的LIB服务124的对应门224。在框308处,门224中适合的门还可以将该调用重定向到被调用的LIB服务124的对应封装器函数226,并将执行控制转移到被调用的LIB服务124的对应封装器函数226。在框310处,被调用的LIB服务124的封装器函数226可以处理该调用,并在LIB上下文222中设置该调用以供被调用的LIB服务124执行,正如早前描述的。在框312处,被调用的LIB服务124的门224可以从LIB上下文222收集该调用的返回值,更新源ISA上下文212,并将执行控制转移到源ISA模拟器202。

[0022] 在图4中框404处(部分300b),在被调用的LIB服务124的过程中或完成时,LIB服务124可以(例如通过调用应用120传递的回调指针来)回调应用120的回调函数125。在框406处,可以根据修改的引用,将执行控制转移到回调函数125的对应封装器函数226。在框408处,封装器函数226可以处理回调,在源ISA上下文212中设置回调以供应用120的回调函数125执行,正如早前描述的,以及此后将执行控制转移到回调函数125的对应门224。在框410处,与回调函数125对应的门224可以将回调重定向到具有封装器函数226准备的源ISA上下文的ISA模拟器,并将执行控制转移到具有封装器函数226准备的源ISA上下文的ISA模拟器。在框412处,源ISA模拟器204可以根据源ISA上下文内的IP来启动回调函数的模拟。在框414处,回调函数125的门224可以从源ISA上下文212收集回调的返回值,更新LIB上下文222,并将执行控制转移到LIB模拟器204以将回调函数125的返回值返回到LIB服务124。

[0023] 本发明的实施例可以与KF结合来利用模拟环境,如图1-4的实施例中任一个实施例,以便提供对内核数据的高效访问同时仍提供对其他函数的模拟。

[0024] 图5包括ISA模拟器504，模拟器504又包括解释器530、解释管理器535和执行器545。解释器530将源ISA指令120解释成目标ISA指令122。当解释器530解释指令时，解释管理器535将这些解释置于解释池560中，其中在解释索引表555中为这些解释建立索引(例如，解释池560中的每个解释按其项地址建立索引)。操作中，执行器545可以通过利用给定IP地址来检索解释，然后执行所解释的指令以遵循应用120的执行流程。正如上文图1-4的多种实施例中描述的，可以通过LIB模拟器506来模拟OS服务。当应用545执行调用源ISA 120中的LIB API的指令时，执行器545将控制转移到LIB模拟器506，LIB模拟器506然后模拟OS服务。可以如上文结合OS使用那样，结合KF使用类似的方法。但是，以此方式处理KF可能导致上下文切换，因为KF尝试访问特权内核区域。此上下文切换可以导致开销。而且，因为通过其IP来识别KF，所以解释器530可能需要跟踪所有KF IP以便识别KF并执行上下文切换。

[0025] 但是，图5在一个实施例中包括KF模拟器540，其提供对内核数据的高效访问同时仍提供用于其他函数的模拟。KF模拟器540包括一组预构建的KF解释541和初始化器542。预构建的解释器541可以利用内核函数的语义来构造。初始化器542与解释管理器535一起工作以确保预构建的KF解释541是可通过常规执行器流程来检索的。

[0026] 例如，在运行时(解释器管理器535将解释索引表555初始化并将解释池560初始化之后)，初始化器542将一个或多个预构建的KF解释器541(例如，KF解释A、B、C)插入到解释池560(稍后可以不一定是KF解释的解释E、F、G、H、I加载解释池560)，并相应地更新解释索引表555。然后，当执行器545利用内核函数地址541(例如，与任何KF解释A、B、C对应的地址)达到IP时，执行器545查询解释管理器535，并从表555和池560获取对应的预构建的解释。然后，执行器545执行解释，其模拟内核函数。

[0027] 利用此实施例，不会产生解释索引表未命中(例如，因为初始化器542较早的动作，那些KF解释已经存在于池560/表555中，不耗费时间来查找未创建且因此不在池560/表555中的KF解释)。再者，此类实施例不再需要解释器530跟踪KF的IP。还免去执行器/解释器上下文切换。例如，对于KF，由于直接调用表555和池560中的KF，所以避开LIB模拟器506。上文任何优点都促成时间模板的KF的效率提高。

[0028] 由此，多个实施例解决了若干缺点。例如，当处理KF以外的代码时，解释器可以从给定的指令指针开始，从该指令指针所指向的存储器读取指令，将指令解码，然后生成目标ISA的指令。解释器然后移到下一个指令指针，并重复该过程。当模拟此类应用(例如，使用结合图2描述的模拟系统)时，将不从内核地址读取该应用。相反，该应用使用内核API来访问内核功能性。但是，在一些平台的情况下，内核地址范围中的一些KF不受保护，并且直接暴露于应用。在没有模拟的情况下，平台可以包含其目标是内核地址的调用指令。因为KF允许访问内核地址，所以这不产生问题。但是，在具有模拟的情况下，可能产生使用相异的源和目标ISA的问题。例如，目标平台可能不以源ISA预期的方式将KF暴露于应用。如果模拟器(例如，PVM)内的解释器尝试读取指令指针(遵循内核函数调用)，则解释器可能无法成功，因为目标平台并未与源ISA所认为的相同来处理KF(例如，KF的存储器地址位置被改变或KF另外以不同的方式实现的)。由此，KF是可以使用上文描述的实施例(例如KF函数模拟器540)适当地处理的特殊实例。在一个实施例中，解释器识别KF地址并以不同于常规非KF函数的方式执行对应解释。例如，对应解释可以使用预构建的解释，如上文描述的。

[0029] 在实施例中,解释管理器535可以“链接”或以其他方式将前一个解释关联到后续解释,所以前一个解释的执行将被传递到后续解释而无需查询解释索引表。此优化能够实现,因为对预构建的解释541进行了准备。再者,在实施例中,将预构建的解释固定在解释池560中,所以垃圾收集不会移除这些解释。

[0030] 返回到图1和图2以及上文论述的实施例,当应用120调用LIB(例如,源LIB)时,(例如通过门224)将呼叫“映射到”目标LIB 124,并如下在目标ISA 122上运行。加载器202加载应用二进制120,并解析应用120中的未定义的符号。ISA模拟器204可以在目标ISA 122的“顶层上”模拟源ISA 120以运行应用120。LIB模拟器206通过映射到目标LIB 124来模拟源LIB。

[0031] 图6包括混合模拟的实施例。一个实施例部分地解释源LIB和部分地映射到目标LIB。例如,加载器602从源LIB 630的源LIB 631加载一些应用120函数以及通过解释以及通过门224映射(例如,如上文结合图1-4描述的)到目标LIB 624来模拟来自源LIB 631的其他应用120。

[0032] 尝试使用此类混合实施例对于如下情况可能有益,例如当目标LIB 635的一些功能性不可用于模拟目的或目标LIB 635不包含与源LIB 630完全相同的语义(由此使模拟困难)。在此类情况下,避免通过目标LIB 635映射(而是直接解释该功能性)可能是有益的。再者,混合模拟在调用(例如调用)源LIB 630函数且解释函数所花的时间少于将相同的函数映射到目标LIB 635并模拟该函数所引入的开销的情况下可以提高应用120性能。

[0033] 但是,混合实施例的简单实现可能产生难题。例如,加载器602可能加载源LIB 630中先前由此类加载的开发者指定的特定库(即,用于直接解释而非用于通过目标LIB 635映射的)。此类库可能包含由于映射开销而直接解释所花时间少于模拟和通过目标LIB 635映射的一些函数(例如,由于门224的使用导致的传输上的延迟)。但是,当加载源ISA LIB 631(以及可能地从LIB 631、671、672等中进行选择)时,这样做可能造成质这样的情形:同一个LIB有两个实例(一个在源ISA中以及另一个在目标ISA中)。源ISA LIB 631的加载还带入全局的额外数据副本。(“全局”是其值能够被程序中位于直接库外的状态或与该全局直接相关的例行程序访问和修改的全局变量。)全局的这些额外副本可能导致源ISA二进制631的不正确的结果,因为其数据值与目标ISA二进制624中的最初数据不同步。例如,一个全局(G)被两个函数(F1和F2)访问。如果F1要被解释,则F1将使用位于源ISA库631中的G副本。如果F2要被映射,则F2将使用位于目标ISA库624中的G副本。这样,如果F1向源ISA库631中的G副本写入,则此更改对于F2将是不可见的(F2使用位于目标ISA库624中的G副本)。如果F2对其相应的G副本进行更改,情况同样如此(所以对G的更改将对于F1是不可见的)。

[0034] 为了解决此潜在问题,图6提供用于同时包含源ISA LIB 631和目标ISA LIB 624的混合环境(例如,一些调用从源库直接解释,而另一些调用从目标库模拟)的实施例。目标LIB 624与源ISA LIB 631共享一些或所有全局(即,全局637)(参见箭头641、642以及匿名全局634、638之间没有对应箭头)。更确切地来说,全局有两种类型:“命名”全局637和“匿名”全局634和638。(命名全局具有与其关联的符号,所以当前库外的程序模块能够引用它们。库代码总是指定对应的符号来引用命名全局。)当加载源ISA库时,加载器将命名全局的符号解析为目标ISA库的存储器地址。但是,匿名全局634、638没有关联的符号(例如,匿名全局可能在C语言程序中包含静态全局)。在将代码编译之后,利用具有当前指令点的偏移

量形式的地址直接访问该全局。

[0035] 图6包括“NoTrans分析器”640,这是分析源ISA二进制120并标识其中不要解释的所有匿名全局638的离线逻辑工具。如果来自应用120的函数使用匿名全局,则该函数被收集在称为“NoTrans函数集”645的函数集中(即,不要解释而是将通过例如门224来映射和模拟的一个函数集合)。如果函数调用NoTrans函数集645中的函数,则该函数本身被收集在NoTrans函数集645中。NoTrans函数集645中的函数不允许被解释(而是被映射)。

[0036] 图7包括有关图6的加载器602的实施例。加载器702将应用720与源ISA库731(用于函数的直接解释)链接,并使用来自ISA门752的门(用于函数的映射和模拟)间接地映射到目标ISA库724。例如,加载器720对应于调度表703,其确定如何链接每个未定义符号。调度表703可以由开发者准备。例如,开发者可以确定需要解释的某些符号,如符号A、E、C是由此指定为用于从源ISA库731解释的那些符号。但是,开发者可以确定其他符号,如符号B、D、F、G、H、I更适于映射,并且相应地指定为用于至目标ISA库724的间接映射。在加载器702加载源ISA应用于720之后,加载器702寻址未定义符号,并尝试通过将该符号与其他库的定义符号关联来解析它们。对于每个符号,如果符号的对应API要解释(调度表703中指定为“要解释”的那些,如符号A、E、C),则将该符号链接到源ISA库731暴露的符号(参见箭头797)。如果符号要映射(根据图7的NoTrans函数集,调度表6中指定为“不要解释”的那些),则将其链接到ISA门752(参见箭头798),ISA门752与目标ISA库724暴露的符号关联。ISA门752可以与ISA模拟器和LIB模拟器一起工作(例如,如结合图1-4解决的),并将源ISA应用120中的函数调用映射到目标LIB 724中的函数实现(参见箭头799)。

[0037] 为了容易地理解,结合一个ISA桥接层123将一个源ISA桥接到一个目标ISA描述了多种实施例。但是,本公开并不局限于此。在多个实施例中,多个ISA桥接层123可以将多个源ISA桥接到一个或多个目标ISA。在这些实施例的其中一些中,可以附加地提供调度器以检测必需的桥接,并且将一个或多个适合的ISA桥接层123实例化以提供必需的ISA桥接。在其他实施例中,用于桥接的一些资源(例如封装器函数的其中一些)可以位于ISA桥接层123可访问的远程服务器上。再者,为了易于理解,将ISA桥接层123描述为配置成将源ISA桥接到目标ISA。但是,对于多种应用,可以采用ISA桥接层123来桥接本身同一个的源ISA和目标ISA。在此类应用中,可能无需所描述的元件的其中一个或多个(例如二进制解释引擎215)。此类应用的示例可以为计算设备102提供增强型工作安全性。其他应用同样可以从此类桥接获益。

[0038] 实施例可以采用代码来实现,并且可以存储在其上存储有指令的存储介质上,这些指令能够用于将系统编程来执行这些指令。该存储介质可以包括但不限于任何类型的存储盘,包括软盘、光盘、光盘、固态硬盘(SSD)、压缩光盘只读存储器(CD-ROM)、可写压缩光盘(CD-RW)和磁光盘;半导体装置,如只读存储器(ROM)、如动态随机存取存储器(DRAM)、静态随机存取存储器(SRAM)的随机存取存储器(RAM)、可擦写可编程只读存储器(EPROM)、闪存存储器、电可擦写可编程只读存储器(EEPROM)、磁卡或光卡;或适于存储电子指令的任何其他类型的介质。本发明的实施例在本文中可参考如指令、函数、过程、数据结构、应用程序、配置设置、代码等来描述。当该数据被机器访问时,该机器可以通过执行任务、定义抽象数据类型、建立低级硬件上下文和/或执行其他操作来进行响应,正如本文更详细描述的。可以将该数据存储在易失性和/或非易失性数据存储装置中。对本公开来说,术语“代码”或

“程序”涵盖宽范围的组件和构造,包括应用、驱动程序、进程、例行程序、方法、模块和子程序。由此,术语“代码”或“程序”可以用于指在被处理系统执行以执行一个或多个期望的操作的任何指令集合。此外,备选实施例可以包括使用比所披露的操作全部更少的过程、使用附加操作的过程、按不同顺序使用相同操作的过程以及其中将本文披露的个体操作组合、拆分或以其他方式改变的过程。在一个实施例中,使用术语控制逻辑包括硬件,如晶体管、寄存器或其他硬件,如可编程逻辑装置。但是,在另一个实施例中,逻辑还包括软件或代码。可以将此类逻辑与硬件集成,如固件或微码。处理器或控制器可以包括要表示本领域中公知的范围广的多种控制逻辑的任何一种的控制逻辑,并且因此,可以较好地作为微处理器、微控制器、场可编程门阵列(FPGA)、专用集成电路(ASIC)、可编程逻辑装置(PLD)等。

[0039] 参考图1,对于一个实施例,可以将处理器和存储器布置104的处理器中至少一个处理器与配置成实施图3-7的方法和实施例的操作(或其子集)的ISA桥接层123的计算逻辑(或其子集)封装在一起。对于一个实施例,可以将处理器和存储器布置104的处理器中至少一个处理器与配置成实施图3 -7的操作(或其子集)和实施例的ISA桥接层123的计算逻辑(或其子集)封装在一起以形成封装中系统(SiP)。对于一个实施例,可以将处理器和存储器布置104的处理器中至少一个处理器与配置成实施图3 -7的操作(或其子集)和实施例的ISA桥接层123的计算逻辑(或其子集)集成在同一个晶片上。对于一个实施例,可以将处理器和存储器布置104的处理器的至少其中之一与ISA桥接层123的计算逻辑(或其子集)集成在同一个晶片上。对于至少一个实施例,该SoC可以被用于桌上型计算机、膝上型计算机、智能电话、计算平板设备、因特网电器、个人数字助理(PDA)、便携式游戏播放设备、服务器或其他计算设备。

[0040] 一个实施例包括一种方法,该方法被至少一个处理器执行,其包括:加载具有源指令集体体系结构(ISA)的应用;将第一源指令的第一解释存储在至少一个存储器,该第一解释是从源ISA到目标ISA;在存储第一解释之后,(a)检索存储的第一解释并执行该解释,以及(b)未成功尝试找到第二源指令的第二解释,以及此后,确定第二解释并执行第二解释。第一源指令可以包括内核函数(KF)。存储第一解释可以包括以具有用于至少一个存储器中存储的解释的索引的格式来存储第一解释。可以将存储的第一解释固定以避免从至少一个存储器移除第一解释。第二解释可以在应用的运行时期间来确定。在一个实施例中,从不模拟第一源指令以及为目标ISA模拟第二源指令。响应用户选择,一种方法可以包括指定用于解释的第一未定义符号以及指定用于模拟的第二未定义符号。一个实施例包括解释第一未定义符号而不模拟以及模拟第二未定义符号。第一和第二未定义符号分别地对应于第一和第二匿名全局变量。一个实施例可以包括通过指向目标ISA库的门来模拟第二未定义符号。一个实施例可以包括与源ISA库共享目标ISA库中存储的命名全局。一个实施例可以包括通过解释来解析第一未定义符号而不模拟,以及通过模拟解析来第二未定义符号。

[0041] 一个实施例可以包括应用,其具有存储在至少一个存储器中的源指令集体体系结构(ISA);至少一个处理器,其具有目标ISA且耦合到至少一个存储器,用于执行包括如下的操作:加载应用;将第一源指令的第一解释存储在至少一个存储器,该第一解释是从源ISA到目标ISA;在存储第一解释之后,(a)检索存储的第一解释并执行该解释,以及(b)未成功尝试找到第二源指令的第二解释,以及此后,确定第二解释并执行第二解释。第一源指令可以包括内核函数(KF)。第二解释可以在应用的运行时期间来确定。实施例可包括解释第一未

定义符号而不模拟以及模拟第二未定义符号。

[0042] 还将认识到，本公开可以是针对为计算设备提供增强型安全性的技术问题的解决方案。本公开的优点可以包括但不限于所提供的隔离的鲁棒性。

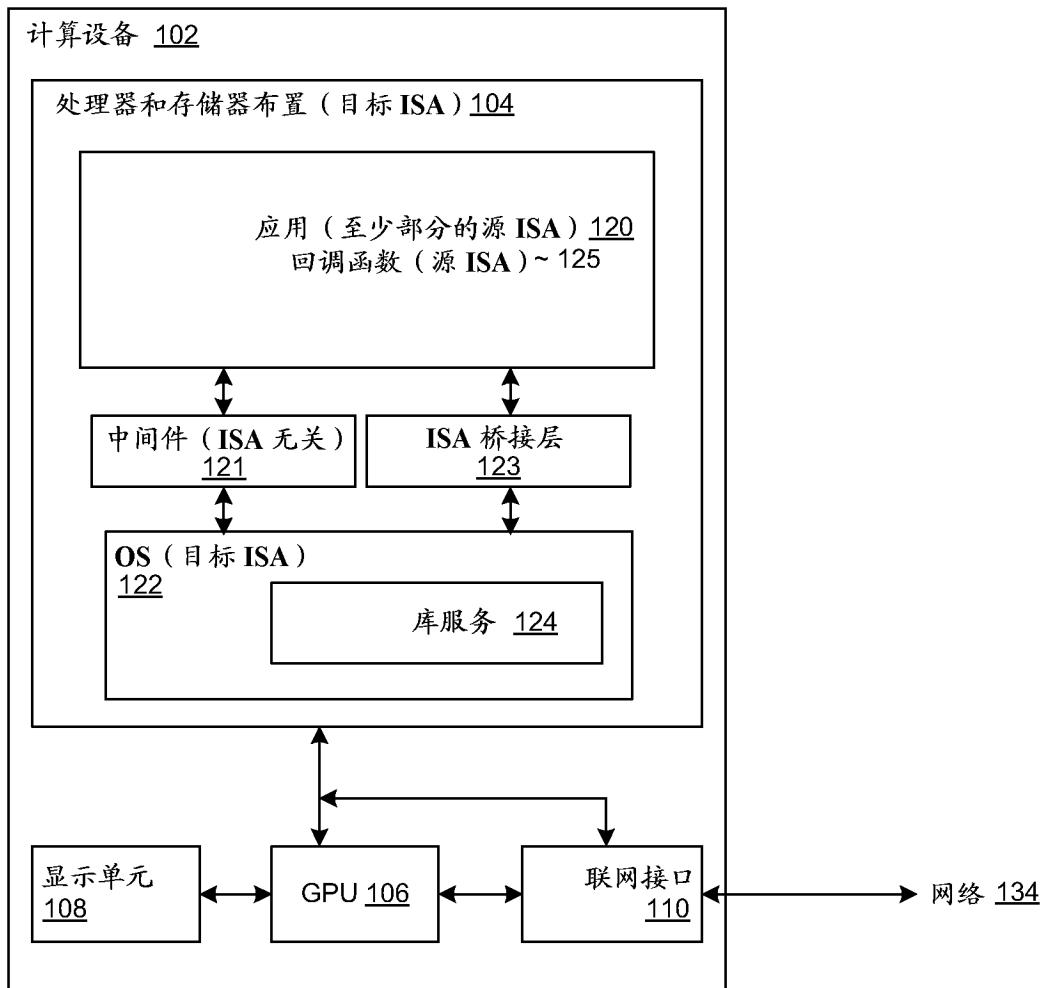


图 1

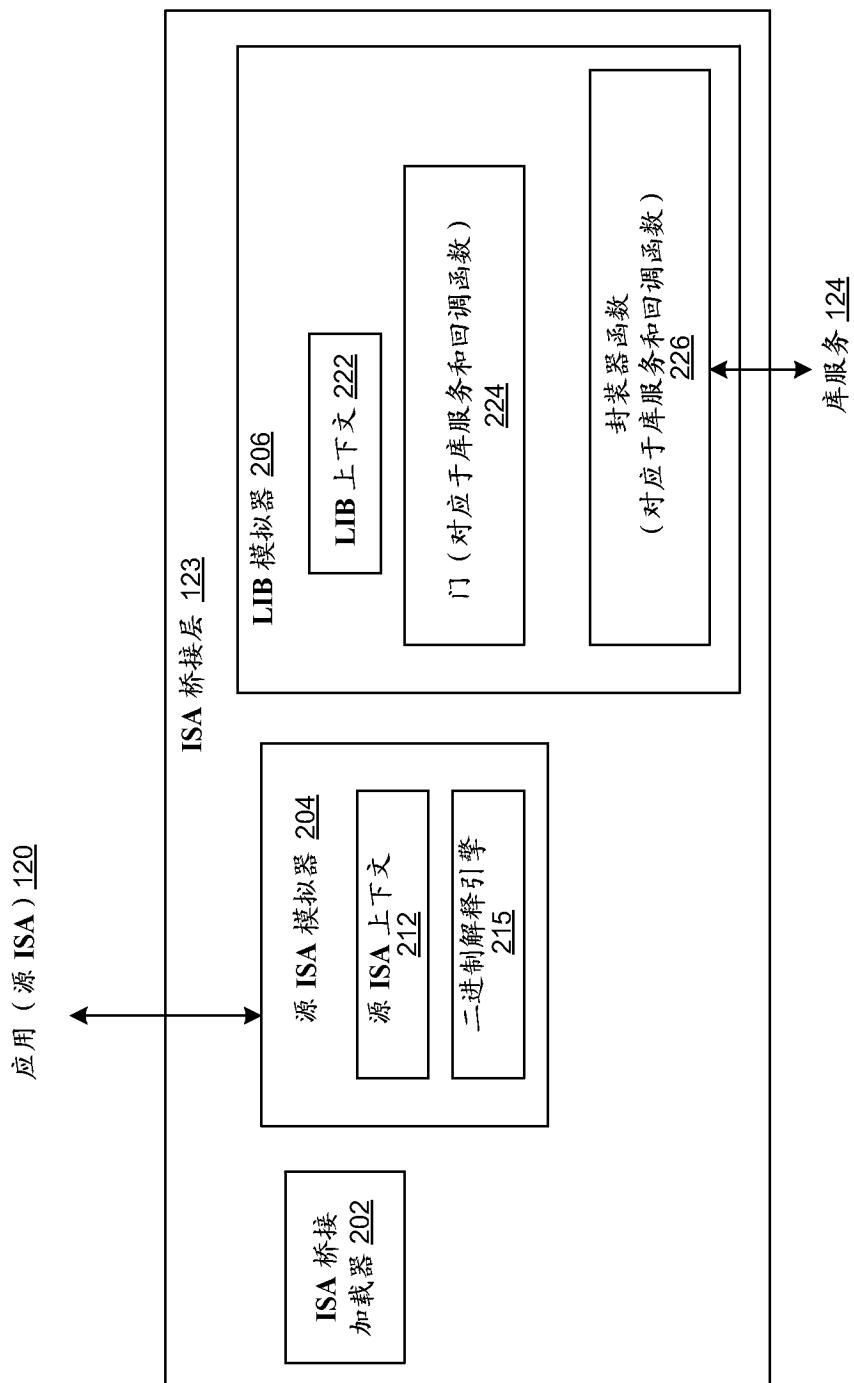


图 2

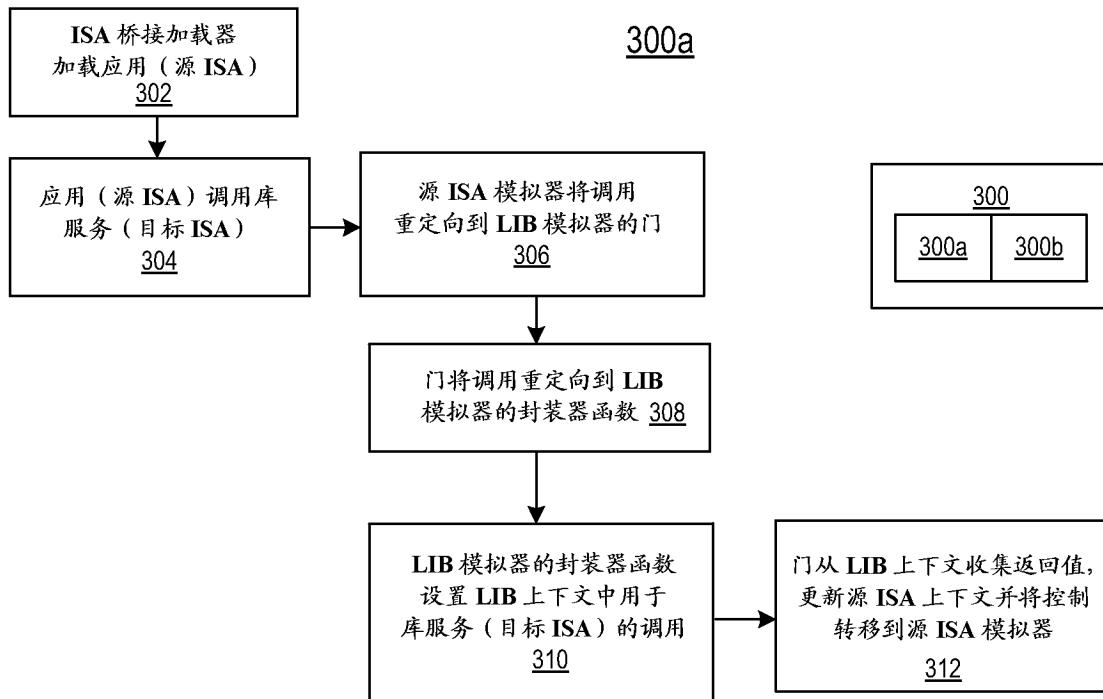


图 3

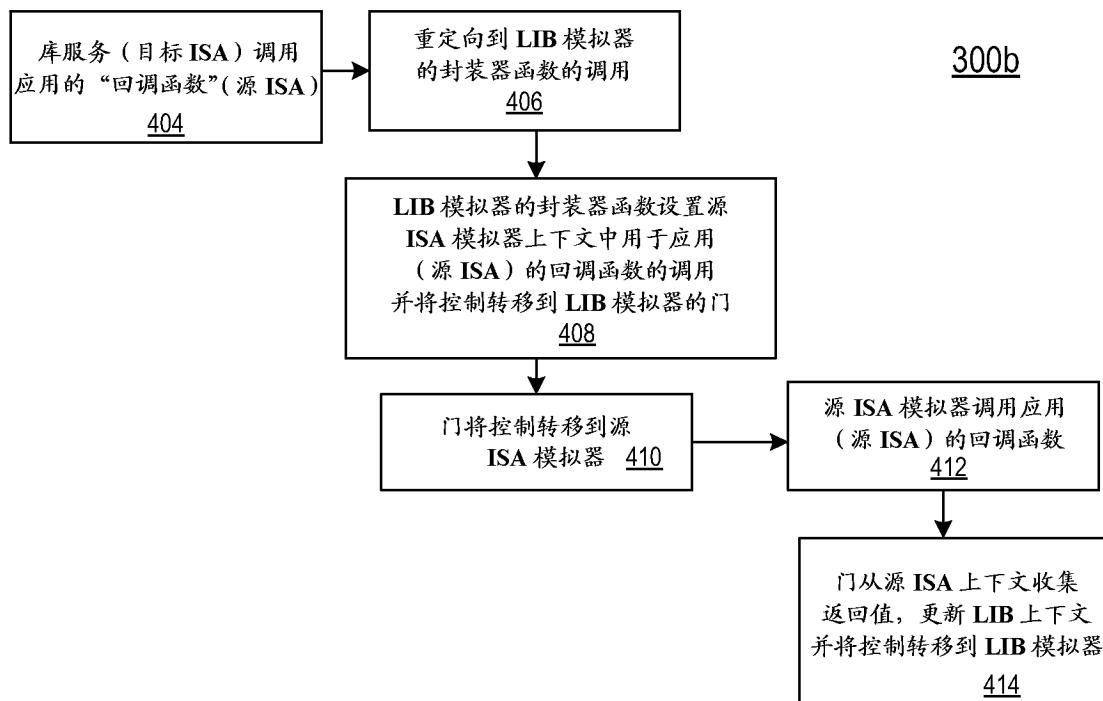


图 4

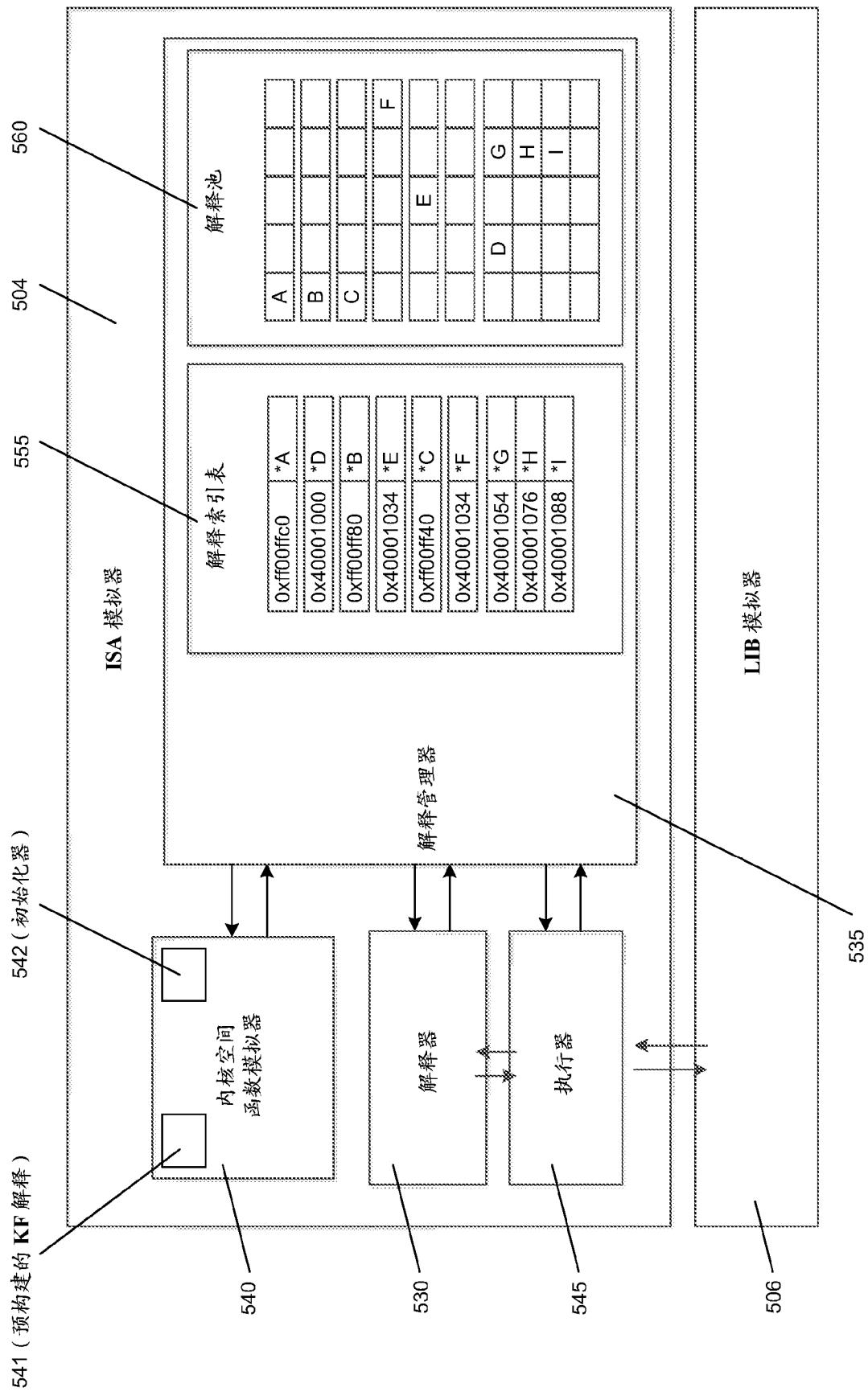


图 51

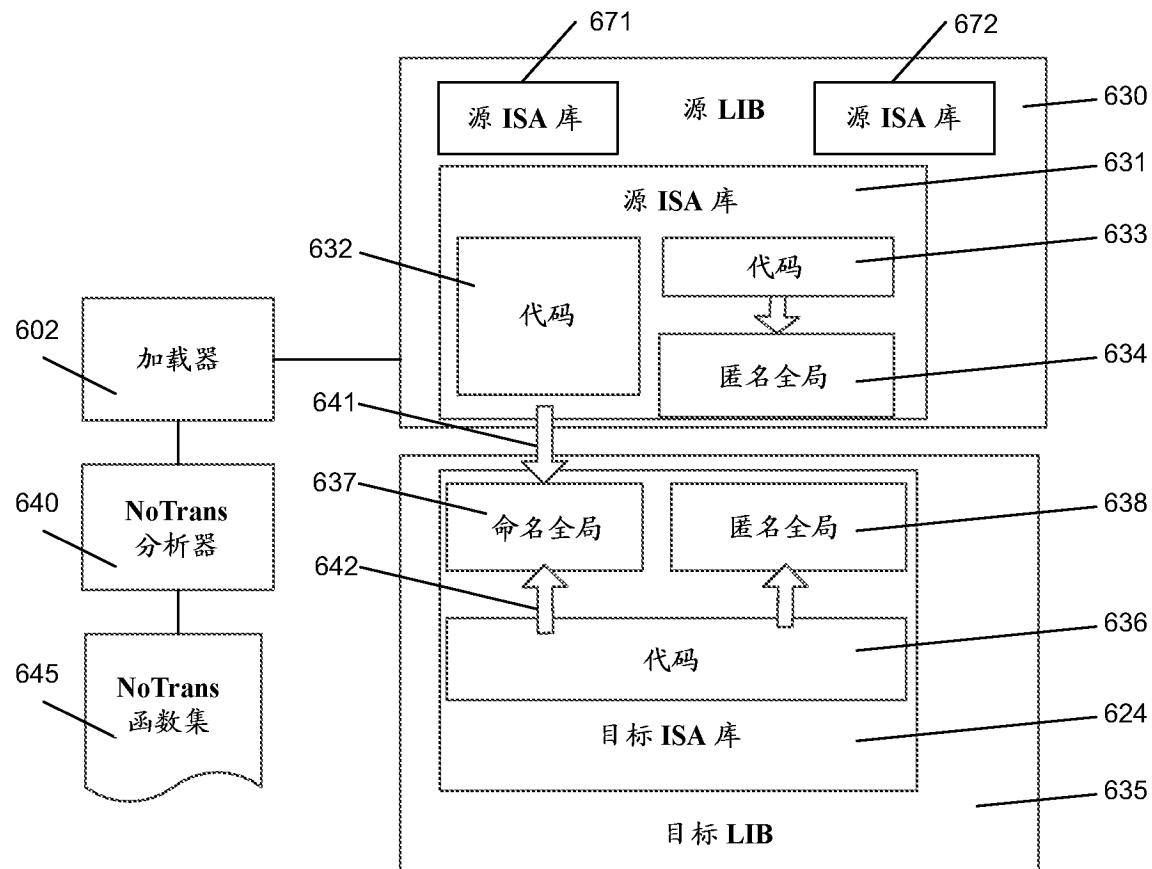


图 6

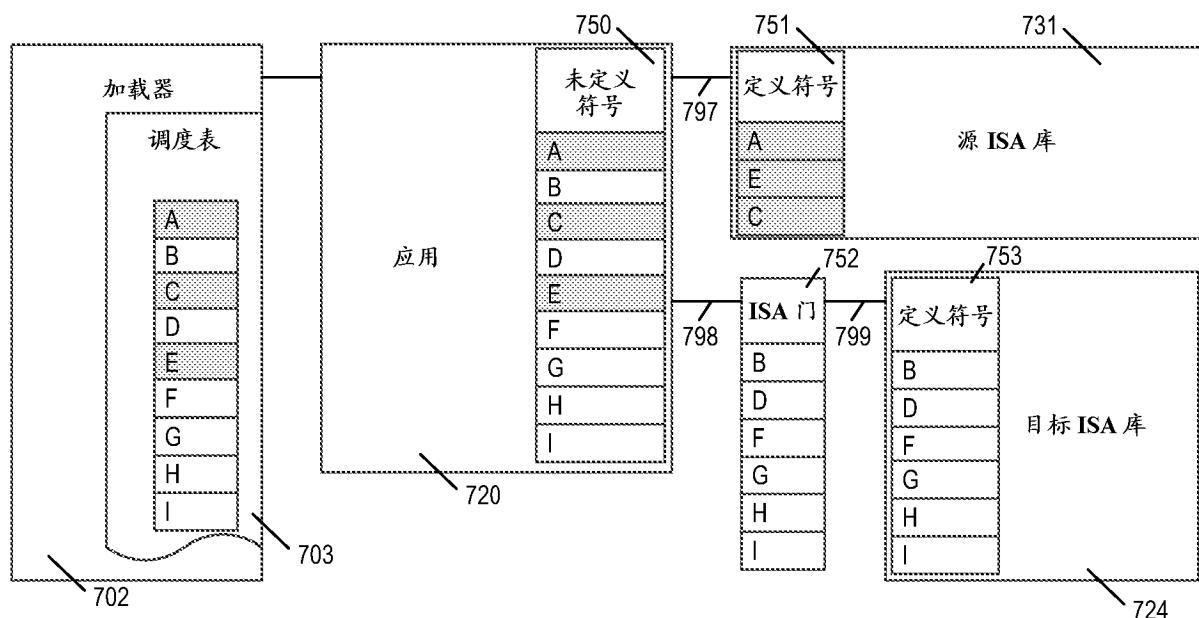


图 7