(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 101553792 B
(45)授权公告日 2013.11.27

(21)申请号 200780045239.9
(22)申请日 2007.11.13
(30)优先权数据
11/635,455 2006.12.06 US
(85)PCT申请进入国家阶段日
2009.06.08
(86)PCT申请的申请数据
PCT/US2007/084522 2007.11.13
(87)PCT申请的公布数据
WO2008/070410 EN 2008.06.12
(73)专利权人 微软公司
地址 美国华盛顿州
(72)发明人 E·P·托奥特 S·甘吉利
R·A·维加
(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 顾嘉运
(51)Int.Cl.
G06F 13/24(2006.01)
(56)对比文件
CN 1564136 A,2005.01.12,
US 2005102671 A1,2005.05.12,
CN 1648866 A,2005.08.03,
审查员 张旭光

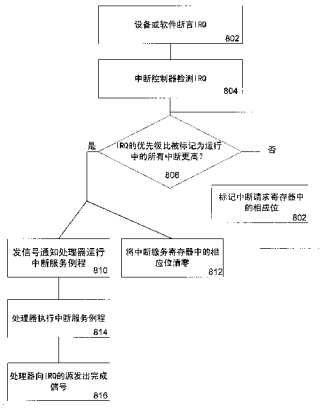
权利要求书1页 说明书10页 附图9页

(54)发明名称

虚拟化环境中的经优化的中断传递

(57)摘要

公开了用于提高虚拟化环境中的中断处理的操作效率的各种操作。一种虚拟化的中断控制器可通过即使在物理中断控制器不提供自动EOI能力时也提供这一机制来避免对显式中断结束命令的需求。对分区间通信使用消息待决位便于避免分区间通信中所使用的处理器间中断的EOI命令,只要对特定消息槽不再提示消息即可。虚拟化的中断控制器即使在其不是最高优先级的运行中中断时也便于中断的选择性EOI,而不管物理中断控制器是否提供这一功能。



CN 101553792 B

1. 一种供虚拟化计算系统处理中断的方法，所述虚拟化计算系统包括虚拟机监控程序和至少一个来宾操作系统，所述方法包括：

在虚拟化的中断控制器处接收第一中断请求；以及

当所述第一中断请求被发送到处理器时立即清除对应于所述第一中断请求的中断服务标志。

2. 如权利要求 1 所述的方法，其特征在于，还包括执行由所述第一个中断规定的动作，并且其中当所述第一中断请求被发送到处理器时立即清除对应于所述第一中断请求的中断服务标志包括在由所述第一个中断规定的动作被完全执行之前清除对应于所述第一中断请求的中断服务标志。

3. 如权利要求 1 所述的方法，其特征在于，还包括：

执行由所述第一个中断规定的动作；以及

准许在执行由所述第一个中断规定的动作的同时传递其他中断。

4. 如权利要求 3 所述的方法，其特征在于，执行由所述第一个中断规定的动作由虚拟机监控程序来仲裁。

5. 如权利要求 1 所述的方法，其特征在于，所述第一中断请求对应于对第一分区间消息的请求。

6. 如权利要求 5 所述的方法，其特征在于，还包括：

提供至少一个消息槽，每一个所述消息槽都与虚拟处理器相关联；以及

在第二分区间消息在所述第一分区间消息的处理完成之前为与所述第一中断请求的处理器相关联的消息槽排队的情况下置位与所述第一分区间消息相关联的消息待决标志。

7. 如权利要求 6 所述的方法，其特征在于，还包括：

在与分区间消息相关联的消息待决位已被置位的情况下在完成处理所述分区间消息后向所述虚拟化的中断控制器发出消息结束命令。

虚拟化环境中的经优化的中断传递

[0001] 背景

[0002] 虚拟机（“VM”）是出于提供仿真机或系统的目的而在计算设备等（例如，主机）上操作的软件构造等。通常，但不必然地，虚拟机是应用程序等，并且可在主机上用于实例化使用应用程序等，同时将这一使用应用程序与该主机设备或该主机设备上的其它应用程序隔离开。在一种典型的情况下，主机可容纳多个部署的 VM，每个 VM 经由可从该主机获得的资源来执行某些预定功能。

[0003] 值得注意的是，尽管采用虚拟的形式，但如主存在计算设备上的每个 VM 无论从哪点看都是计算机器，并因此将其本身如此向其使用应用程序和外部世界两者表示。作为示例，VM 和 / 或其使用应用程序能够并且实际上的确对该 VM 的硬件资源发出硬件请求，即使该 VM 可能实际上不具有这些硬件资源。相反，且可以理解，这些硬件请求被截取或以其他方式被重定向到主机，并且这一主机通常在发出请求的 VM 和 / 或其使用应用程序仍旧一无所知的情况下基于其硬件资源来为这些硬件请求服务。

[0004] 通常，尽管并非必须，但主机将其每一个 VM 部署在单独的分区、地址空间、处理区域等中。这一主机可包括具有担当监督应用程序或“管理程序”的虚拟机监控程序（“VMM”）等的虚拟化层，其中虚拟化层监督和 / 或以其它方式管理该主机的每一 VM 的管理方面，并担当每一 VM 和外界之间的可能链接。VMM 可以是在其自己的地址空间中运行的单独的应用程序或者可以与主机操作系统更紧密地集成，或者直接地或者作为某一种操作系统扩展，诸如设备驱动程序等。值得注意的是，主机的 VMM 可截取或以其他方式重定向源自该主机的每一个 VM 和 / 或其使用应用程序的硬件请求，并且可同样在发出请求的 VM 和 / 或其使用应用程序仍旧一无所知的情况下至少帮助服务这些请求。

[0005] 许多计算系统包括多个处理器。多处理器虚拟机环境中的处理器能以来宾模式或 VMM 模式操作。当以来宾模式运行时，处理器使用虚拟机定义管理虚拟机的来宾操作系统和应用程序，从而在没有来自 VMM 的介入的情况下转换自变量并管理系统资源。有时，来宾操作系统或应用程序可能需要必须由 VMM 管理的系统资源。作为示例，VMM 对于出错处理、系统错误或中断处理可能是必需的。在这些情况下，处理器以 VMM 模式操作。

[0006] 现代处理系统包括对允许将外部事件通知给处理器的中断的支持。例如，当用户按压键盘上的键或网络分组通过线缆到达时，生成相应的中断并将其发送到处理器。通常，中断使得处理器停止其正在做的事情，记录其当前执行位置以使其能够在为该中断服务并且然后执行指定的中断服务例程后恢复执行。

[0007] 计算系统可包括引导并仲裁系统中的中断流的一个或多个中断控制器。中断控制器逻辑可以用分立的硬件组件来实施，可被集成到处理器中，或可被虚拟化。中断控制器特别地负责确定中断的优先级以及将中断定向到多处理器环境中的适当处理器。在虚拟化环境中，处理器和中断控制器可被虚拟化。这一般通过诸如虚拟机监控程序等软件以及由硬件提供的虚拟化辅助装置 (assist) 的组合来实现。

[0008] 一般而言，在处理了中断后，经由中断结束 (EOI) 命令来通知中断控制器。该命令告诉中断控制器现在可以传递其传递可能已经在处理前一中断时被推迟的其他中断。EOI

命令通常通过I/O端口或诸如对寄存器的读写等存储器映射的I/O访问来被传递给中断控制器。对于物理中断控制器而言，处理EOI命令可能消耗数十或数百个周期。对于虚拟化的中断控制器而言，处理EOI命令可能消耗数千个周期。

[0009] 某些虚拟机监控程序将中断用作分区间消息传送的基础。如果在一个分区中运行的软件需要与在同一物理机器上的第二分区中运行的软件进行通信，则它能够通过使用分区间消息来这样做。当消息由一个处理器来发送时，虚拟机监控程序可向作为该消息的预期接收者的处理器发送中断，从而使得该接收者处理器的中断服务例程处理该消息并响应其内容。

[0010] 概述

[0011] 提供本概述是为了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。该概述不旨在标识所要求保护的主题的关键特征或必要特征，也不旨在用于帮助确定所要求保护的主题的范围。

[0012] 此处描述了用于在多处理器虚拟化环境中高效地处理中断的机制。在某些实施例中，来宾操作系统可将特定中断源编程为“自动中断结束”（“自动EOI”）。在处理自动EOI时，虚拟化的中断控制器将中断服务寄存器中对应于所传递的中断的位清零而不等待显式中断结束（“EOI”）命令。自动EOI中断可不阻塞其他中断的传递。

[0013] 中断可用于实现分区间通信。当来宾操作系统接收到与分区间消息相关联的中断时，该来宾操作系统的中断服务例程从指定的消息槽中读取该消息并基于该消息类型和净荷来执行动作。如以下详细描述的，来宾操作系统可通过仅在另一消息对同一消息槽排队的情况下在完成消息处理后发送显式消息结束（“EOM”）命令来消除在处理分区间消息时招致的某些开销。EOM命令的计算成本大致等同于EOI的计算成本，但EOM仅在其他消息在排队的罕见情况下发送。这可显著地降低用于分区间通信的中断处理的平均成本。

[0014] 中断可具有各种优先级。一般而言，较高优先级的中断可打断较低优先级的中断的处理，但反之却不然。在虚拟化环境中，对于来宾操作系统而言发出针对不是最高优先级的运行中的中断的中断的EOI命令是可能的。此处描述了虚拟化的中断控制器用于检查传入EOI命令是否对应于最高优先级的运行中的中断的机制。如果不是，则经虚拟化的中断控制器将传入EOI命令的中断向量添加到需要在稍后被EOI的一组中断。如果传入EOI命令的确对应于最高优先级的运行中的中断，则经虚拟化的中断控制器不仅处理针对相应中断的EOI命令而且处理对应于先前为稍后EOI标记的所有其他中断的EOI命令。

[0015] 附图简述

[0016] 图1是表示计算机系统中用于经虚拟化操作环境的硬件和软件体系结构的逻辑分层的框图；

[0017] 图2是表示其中虚拟化由主机操作系统（或者直接地或者经由管理程序）来执行的经虚拟化计算系统的框图；

[0018] 图3是表示其中虚拟化由与主机操作系统并排运行的虚拟机监控程序执行的替换的经虚拟化计算系统的框图；

[0019] 图4是表示其中虚拟化由独立于主机操作系统运行的虚拟化器执行的另一替换的经虚拟化计算系统的框图；

[0020] 图5是具有中断控制器的计算系统的一部分的框图；

[0021] 图 6 是示出处理中断请求的方法的流程图；

[0022] 图 7 描绘了用于中断优先级的示例的时间线；

[0023] 图 8 是示出根据此处示教的使用自动 EOI 来处理中断请求的方法的流程图；以及

[0024] 图 9 是示出根据此处示教的处理处理器间消息的方法的流程图。

[0025] 详细描述

[0026] 在以下描述和附图中描述了某些具体细节，以提供对本发明的各个实施例的全面理解。通常与计算和软件技术相关联的某些公知细节将不在以下公开中描述，以避免不必要地使本发明的各实施例晦涩难懂。此外，相关领域的普通技术人员可以理解，他们可以无需以下描述的细节中的一个或多个而实现其它实施例。最后，尽管在以下公开中参考了步骤和序列来描述各个方法，但是如此的描述是为了提供本发明的实施例的清楚实现，且步骤以及步骤序列不应被认为是实现本发明所必需的。

[0027] 应该理解，此处描述的各种技术可以结合硬件或软件，或在适当时以两者的组合来实现。因此，方法和装置或其某些方面或部分，可以采用包含在诸如软盘、CD-ROM、硬盘驱动器或任何其它机器可读存储介质等有形介质中的程序代码（即，指令）的形式，其中，当程序代码被加载至诸如计算机等机器并由其运行时，该机器成为用于实现本发明的装置。在程序代码在可编程计算机上执行的情况下，计算设备通常包括处理器、该处理器可读的存储介质（包括易失性和非易失性的存储器和 / 或存储元件）、至少一个输入设备、以及至少一个输出设备。一个或多个程序可以例如，通过使用 API、可重用控件等来实现或利用结合本发明描述的过程。这样的程序优选地用高级过程语言或面向对象编程语言来实现，以与计算机系统通信。然而，如果需要，程序可以用汇编语言或机器语言来实现。在任何情形中，语言可以是编译的或解释的语言，且与硬件实现相结合。

[0028] 尽管示例性实施例可涉及在一个或多个独立计算机系统的上下文中利用本发明的各方面，但本发明不受此限制，而是可以结合任何计算环境，诸如网络或分布式计算环境来实现。而且，本发明的各方面可以在多个处理芯片或设备中实现或跨多个处理芯片或设备实现，且存储可以类似地跨多个设备来实现。这样的设备可以包括，个人计算机、网络服务器、手持式设备、超级计算机或集成在诸如汽车和飞机等其它系统中的计算机。

[0029] 概览

[0030] 描述了用于在虚拟机环境中高效地处理中断的各方法和系统。中断出于各种目的而在现代计算系统中使用，包括作为示例，以将外部事件通知给处理器以及便于多处理器系统的各处理器之间的通信。通常，中断打断普通处理并临时将控制流转向中断服务例程（“ISR”）。计算系统的各种活动可触发中断。某些示例是按压键盘上的键、接收网络分组以及对盘进行读写。处理器间中断（“IPI”）是一个处理器可用于打断多处理器环境中的另一处理器的中断类型。IPI 可用作处理器间消息传送的基础。

[0031] 计算系统通常包括引导并仲裁系统中的中断流的一个或多个中断控制器。中断控制器负责区分传入中断的优先顺序并将其定向到多处理器系统中的适当处理器。中断控制器可以用硬件来实现并由此可以是分立组件或可与处理器集成。中断控制器还可被虚拟化。这通常通过软件以及由硬件提供的虚拟化辅助装置的组合来实现。软件可以是执行与物理中断控制器相同的基本功能的虚拟机监控程序的一部分。

[0032] 通常，每一中断源都具有指定的中断优先级。作为一个示例，这些优先级可被标号

为 1 到 255，255 为最高优先级而 0 为最低优先级。较高优先级的中断被允许打断正在处理较低优先级中断的中断服务例程，但较低优先级的中断不被允许打断较高优先级的中断。当中断服务例程完成执行时，它在其上运行的处理器通常发出 EOI 命令，从而发信号通知中断控制器该中断的处理已完成并且现在可传递被推迟的较低优先级的中断。

[0033] 对于虚拟化的中断控制器，EOI 命令用执行与将响应该 EOI 命令的物理中断控制器相同的操作的软件来实现。这通常涉及截取对 EOI 端口或寄存器的访问并调用软件处理程序。截取和软件处理程序的组合通常需要数千或数万个周期来处理 EOI 命令，从而对虚拟化环境中的中断服务例程的操作增添了显著的开销。

[0034] 此处所描述的方法和系统提供了用于高效地处理中断的机制。能够在许多情况下跳过 EOI 命令，从而显著地减少与中断传递相关的虚拟化开销。在用于处理器间消息传送的 IPI 的情况下，消息结束 EOI 仅在第二消息已经为包含刚刚处理的消息的槽排队时才需要被发送。在某些情况下，可选择性地 EOI 物理中断而不管其是否是最高优先级的运行中中断。

[0035] 虚拟化概述

[0036] 操作系统和处理器指令集的多样性可导致软件的互操作性降低。高级语言和操作系统两者中的存储器和 I/O 抽象可去除某些硬件资源依赖，但某一些仍然保留。许多操作系统都是针对特定系统体系结构来开发的并被设计成直接管理硬件资源。这可限制计算机系统在可用软件和操作系统方面的灵活性并可负面地影响安全性和故障隔离，尤其在系统由多个用户共享时。

[0037] 虚拟化提供了用于在增强安全性和可靠性的同时增加灵活性的机制。处理器、存储器和 I/O 设备是可被虚拟化的子系统的示例。当一子系统被虚拟化时，将虚拟接口和通过该虚拟接口可用的虚拟资源映射到在其上实现该虚拟化的真实系统的接口以及资源。虚拟化可不仅应用于子系统而且应用于整个机器。虚拟机的体系结构在真实机器上的软件层中实现。

[0038] 从概念的观点来看，计算机系统一般包括在基本硬件层上运行的一个或多个软件层。该分层出于各抽象原因而完成。通过为给定软件层定义接口，该层可由其上的其他层不同地实现。在设计良好的计算机系统中，每一层只知道（并且只依赖于）紧接在其下的层。这允许在不负面地影响层或“栈”（多个邻接层）上的各层的情况下替换所述层或栈。例如，软件应用程序（上层）通常依赖下层的操作系统（下层）将文件写入某种形式的永久存储，并且这些应用程序不需要明白将数据写入软盘、硬盘或网络文件夹之间的区别。如果该下层用用于写入文件的新操作系统组件来替换，则上层软件应用程序的操作保持不受影响。

[0039] 分层软件的灵活性允许虚拟机 (VM) 呈现实际上是另一软件层的虚拟硬件层。以此方式，VM 可为其上的软件层创建所述软件层正在其自己的专用计算机系统上运行的幻觉，并由此 VM 能够允许多个“来宾系统”同时在单个“主机系统”上运行。

[0040] 图 1 是表示计算机系统中用于虚拟化环境的硬件和软件体系结构的逻辑分层的框图。在图 1 中，虚拟化程序 110 直接或间接地在物理硬件体系结构 112 上运行。虚拟化程序 110 可以是 (a) 与主机操作系统并排运行的虚拟机监控程序或 (b) 具有执行虚拟化的管理程序组件的主机操作系统。术语虚拟机监控程序用作对于各种类型的虚拟化程序中的

任一种的通用术语。虚拟化程序 110 虚拟化来宾硬件体系结构 108(被示为虚线以示出该组件是分区或“虚拟机”的事实),即,实际上并不存在而是由虚拟化程序 110 虚拟化的硬件。来宾操作系统 106 在来宾硬件体系结构 108 上执行,而软件应用程序 104 能够在来宾操作系统 106 上执行。在图 1 的虚拟化操作环境中,即使软件应用程序 104 被设计成在一般不与主机操作系统和硬件体系结构 112 兼容的操作系统上运行,该软件应用程序 104 也可在计算机系统 102 中运行。

[0041] 接着,图 2 示出了包括直接在物理计算机硬件 202 上运行的主机操作系统(主机 OS)软件层 204 的虚拟化计算系统,其中主机 OS 204 通过向分别供操作系统 A 和 B(即,212 和 214)使用的分区 A 208 和 B 210 展示接口来提供对物理计算机硬件 202 的资源的访问。这使得主机 OS 204 能够不被在其上运行的操作系统层 212 和 214 注意。同样,为了执行虚拟化,主机 OS 204 可以是具有本机虚拟化能力的特别设计的操作系统或另选地,它可以是具有用于执行虚拟化的结合的管理程序组件(未示出)的标准操作系统。

[0042] 再次参考图 2,主机 OS 204 之上是两个分区,即,分区 A 208 和分区 B 210,前者可以是例如虚拟化的 Intel 386 处理器,而后者可以是例如摩托罗拉 680X0 处理器系列中的一个的虚拟化版本。在每一个分区 208 和 210 中的分别是来宾操作系统(来宾 OS)A 212 和 B 214。在来宾 OS A 212 的上方运行的是两个应用程序,即,应用程序 A1 216 和应用程序 A2 218,而在来宾 OS B 214 上运行的是应用程序 B1 220。

[0043] 对于图 2,注意分区 A 208 和分区 B 214(用虚线示出的)是只作为软件构造存在的虚拟化计算机硬件表示是重要的。它们由于专用虚拟化软件的执行而成为可能,该专用虚拟化软件不仅只将分区 A 208 和分区 B 210 分别呈现给来宾 OS A 212 和来宾 OS B 214,而且还执行来宾 OS A 212 和来宾 OS B 214 间接地与真实物理计算机硬件 202 交互所需的所有软件步骤。物理计算机硬件 202 可包括如单处理器环境中的单个中央处理单元(CPU)222,或如多处理器环境中的多个 CPU 222、224、226。

[0044] 图 3 示出了其中虚拟化由与主机操作系统 306 并排运行的 VMM 304 执行的替换虚拟化的计算系统。在某些情况下,VMM 304 可以是在主机操作系统 306 上运行并仅通过主机操作系统 306 来与计算机硬件 302 交互的应用程序。在其他情况下,诸如图 3 所示,VMM 304 可替代地包括部分独立的软件系统,其在某些层上经由主机操作系统 306 来间接地与计算机硬件 302 交互,但在其他层上 VMM 304 直接与计算机硬件 302 交互(类似于主机操作系统直接与计算机硬件交互的方式)。并且在又一些情况下,VMM 304 可包括完全独立的软件系统,其在所有层上直接与计算机硬件 302 交互(类似于主机操作系统直接与计算机硬件交互的方式)而不利用主机操作系统 306(但仍旧与主机操作系统 306 交互以协调对计算机硬件 302 的使用并避免冲突等)。

[0045] 在图 3 所示的示例中,两个分区,即,A 308 和 B 310,在概念上位于 VMM304 之上。在每一个分区 308 和 310 中的分别是来宾操作系统(来宾 OS)A 312 和 B 314。在来宾 OS A 312 的上方运行的是两个应用程序,即,应用程序 A1 316 和应用程序 A2 318,而在来宾 OS B 314 上运行的是应用程序 B1,320。物理计算机硬件 302 可包括如单处理器环境中的单个中央处理单元 (CPU)322,或如多处理器环境中的多个 CPU 322、324、326。

[0046] 图 4 示出了其中虚拟化由管理程序 404 执行的另一替换的虚拟化计算系统。管理程序 404 包括可以直接与计算机硬件 402 交互而不使用主机操作系统的独立软件系统。物

理计算机硬件 402 可包括如单处理器环境中的单个中央处理单元 (CPU)422，或如多处理器环境中的多个 CPU 422、424、426。

[0047] 在图 4 所示的示例中，两个分区，即，A 408 和 B 410，在概念上位于 VMM404 之上。在每一个分区 408 和 410 中的分别是来宾操作系统（来宾 OS)A 412 和 B 414。在来宾 OS A 412 的上方运行的是两个应用程序，即，应用程序 A1 416 和应用程序 A2 418，而在来宾 OS B 414 上运行的是应用程序 B1 420。来宾 OS A 412 提供主机 OS 服务。物理计算机硬件 402 可包括如单处理器环境中的单个中央处理单元 (CPU)422，或如多处理器环境中的多个 CPU 422、424、426。

[0048] 用于实现以上提到的分区的所有这些变体仅仅是示例性实现，并且此处没有一样应被解释为将本发明限于任何特定虚拟化方面。

[0049] 中断处理概述

[0050] 图 5 是具有中断控制器的多处理器计算系统的一部分的说明性示例的框图。任何数量的设备 502、504、506、508 都可用作中断请求的源。设备 502、504、506、508 可以是物理设备，诸如例如，键盘、盘驱动器、网卡，或者可以是虚拟化设备。中断请求还可由处理器 510、512、514 中的任一个来生成。

[0051] 中断控制器 516 仲裁并指导中断请求的处理。中断控制器 516 可以是物理设备，诸如可编程中断控制器（“PIC”）或高级可编程中断控制器（“APIC”）。或者，中断控制器 516 可被虚拟化，在这种情况下其功能由诸如 VMM 中的软件处理程序等软件来执行。

[0052] 大多数中断控制器跟踪所请求的且运行中的中断请求。这通常通过使用其中每一个位表示单独的中断源的两个位向量来完成。一个位向量被称为中断请求寄存器 518，而第二个被称为中断服务寄存器 520。当中断控制器 516 接收到对一中断的请求时，它设置中断请求寄存器 518 中的相应的位。当中断控制器 516 将中断传递给处理器 510、512 或 514 时，它将中断请求寄存器 518 中的相应位清零并在中断服务寄存器 520 中设置相应的位。当中断控制器 516 接收到 EOI 时，它知道不再为相应的中断服务并因此将中断服务寄存器 520 中的相应位清零。此时，中断控制器 516 扫描中断请求寄存器 518 以确定尚未被服务的最高优先级的所请求中断。如果这一中断的优先级高于最高优先级的运行中中断，则中断控制器打断该较低优先级的运行中中断的中断服务例程。

[0053] 中断请求的传统生存周期在图 6 中描绘。设备或软件通过断言中断请求（“IRQ”)602 来开始该过程。中断请求可由各种各样的源来生成。源可包括，作为示例且并非出于限制的目的，键盘、鼠标、声卡、调制解调器、通信端口、定时设备和软件指令。在“电平触发”中断中，希望发信号通知中断的设备将中断请求线路上的电压驱动到被定义为“活动”的预定水平并将维持该电压水平直到该中断被服务。在“边缘触发”中断中，中断请求通过中断请求线路上的电平转换来发信号通知，其中希望发信号通知中断的设备将脉冲驱动到该中断请求线路上并且然后将该线路返回到其静止状态。

[0054] 一旦中断控制器检测到 IRQ 604，它就可能通过检查中断服务寄存器 520（图 5）来确定该 IRQ 是否具有比任何当前运行中中断更高的优先级 606。如果当检测到 IRQ 时较高优先级的中断正在运行中，则中断控制器标记中断请求寄存器中的相应位 608 以记录该待决请求。如果所请求的中断具有比任何运行中的中断更高的优先级，则中断控制器标记中断服务寄存器中的相应位 610 并发信号通知适当的处理器运行相应的中断服务例程 612。

在完成中断服务例程的执行 614 后，该处理器通常可通过对 I/O 端口或存储器映射的寄存器进行读写来向中断设备指示该中断已被处理 616。该处理器通过发送 EOI 命令 618 来发信号通知中断控制器该中断已被处理，EOI 命令通常通过 I/O 端口或诸如对寄存器的读写等存储器映射的 I/O 访问来传递。中断控制器处理该 EOI 命令并清除中断服务寄存器中的相应标志 620。EOI 命令告诉中断控制现在可传递已被推迟的较低优先级的中断。例如，如果网络分组到达从而触发一中断，则当较高优先级的键盘中断正被服务时，该网络分组中断请求可能已由中断控制器保持待决直到该较高优先级的键盘中断被完全服务。处理 EOI 命令对于物理中断控制器而言可能花费数十或数百个周期或对于虚拟化的中断控制器而言甚至可能花费数千个周期。

[0055] 图 7 描述了示出中断优先级的一般概念并且不旨在是限制性的示例。假设在时刻 t_1 处优先级为 10 的中断源请求一中断 702。中断控制器打断调用与该中断源相关联的中断服务例程 704 的处理器。现在假设在用于优先级为 10 的中断的 ISR 完成处理之前的时刻 t_2 处，优先级为 200 的中断源请求发往同一处理器的中断 706。中断控制器再次打断处理器，并且用于优先级为 200 的中断的 ISR 708 开始执行，而用于优先级为 10 的中断的 ISR 被挂起 710。假设优先级为 50 的第三中断源在优先级为 200 的 ISR 708 完成之前的时刻 t_3 处请求中断 712。中断控制器将推迟该中断的传递 714 直到该处理器在时刻 t_4 处完成优先级为 200 的 ISR 的执行。优先级为 50 的 ISR 将在优先级为 200 的 ISR 708 完成之后调用 716。当在时刻 t_5 处优先级为 50 的 ISR 716 完成时，优先级为 10 的 ISR 的执行恢复 718 并且在时刻 t_6 处完成。

[0056] 虚拟化的中断控制器

[0057] 在虚拟化环境中，处理器和中断控制器可被虚拟化。这通过软件（例如，VMM）和由硬件提供的虚拟化辅助装置的组合来完成。在典型的安排中，EOI 命令由 VMM 来仿真。这通过截取对 EOI 端口或寄存器的访问来完成。该截取调用 VMM 中响应该 EOI 执行与物理中断控制器相同的功能的软件处理程序。该截取和软件处理程序的组合可能需要数千或数万个周期。这当在虚拟化环境中执行时为 ISR 增添了显著的开销。

[0058] 通常，VMM 接受中断并将其作为虚拟中断重定向到来宾操作系统。中断可从各种源中生成，包括，作为示例而非限制，物理硬件设备、仿真硬件设备的分区、希望向另一分区发送消息或发信号通知事件的分区，或希望发信号通知分区的 VMM。VMM 通常在中断已被接受之后向物理中断控制器发出 EOI 命令。对于电平触发中断而言，直到来宾操作系统中的 ISR 已运行并向虚拟中断控制器发出 EOI 命令才发出 EOI 命令一般是不安全的。如 APIC 的某些物理中断控制器仅允许 EOI 最高优先级的运行中中断。然而在某些情况下，VMM 可能需要选择性地 EOI 不是最高优先级的运行中中断的中断。作为例示，考虑当其中第一个是较低优先级的两个电平触发中断相继到达时会发生什么。VMM 接受第一个中断并将其重定向到来宾操作系统。在该来宾操作系统中的 ISR 发出 EOI 命令之前，第二个中断到达。随后该来宾操作系统发出对第一个中断的 EOI 命令。在这种情况下，VMM 无法 EOI 第一个中断，因为较高优先级的中断已经在运行中并且向物理中断控制器发出 EOI 命令将会 EOI 该较高优先级的中断。

[0059] 根据此处的本发明，来宾操作系统被允许将某些中断源编程为“自动 EOI”。当中断源被标记为自动 EOI 时，修改传统中断优先化行为。自动 EOI 中断不阻塞其他中断的传

递。因此，实际上，自动EOI中断类似于最低优先级的中断那样工作，因为包括其他自动EOI中断在内的任何其他中断被允许打断其相关联的ISR的执行。

[0060] 在传递自动EOI中断时，中断服务寄存器中与该自动EOI中断相关联的位被立即清零。实际上，虚拟化的中断控制器在传递自动EOI中断时自动生成EOI。对于自动EOI中断而言，中断源通过直到其知道前一中断已被处理才请求后续中断来缓和(moderate)其本身是合乎需要的。否则，每一个后继中断都将打断前一ISR，从而可能溢出处理器的栈。

[0061] 在一个实施例中，自动EOI属性在与合成中断源(SINT)相关联的虚拟寄存器中指定。该虚拟寄存器的格式如下：

	位	描述	属性
[0062]			
[0063]	63:18	RsvdP(值应当被保留)	读/写
[0064]	17	自动EOI	读/写
[0065]		在隐式EOI应在中断	
[0066]		传递后执行的情况下置位	
[0067]	16	在SINT被屏蔽的情况下置位	读/写
[0068]	15:8	RsvdP(值应当被保留)	读/写
[0069]	7:0	向量	读/写

[0070] 在虚拟处理器创建时间，所有SINT寄存器的默认值是0x0000000000010000。因此，所有合成中断源默认都是被屏蔽的。来宾操作系统必须通过对适当的向量编程并将位16清零来去掉其屏蔽。

[0071] 自动EOI标志指示隐式EOI在中断被传递给虚拟处理器时应由VMM来执行。另外，VMM将自动清除虚拟中断控制器的运行中寄存器内的相应标志。如果来宾操作系统允许该行为，则它必须不在其中断服务例程中执行显式EOI。

[0072] 图8描绘了根据此处的本发明的自动EOI中断请求的生存周期。设备或软件通过断言IRQ 802来开始该过程。一旦中断控制器检测到IRQ 804，它就可能通过检查中断服务寄存器520(图5)来确定该IRQ是否具有比当前被标记为运行中的任何中断更高的优先级806。如果当检测到IRQ时较高优先级的中断被标记为运行中，则中断控制器标记中断请求寄存器中的相应位808以记录该待决请求。如果所请求的中断具有比被标记为运行中的任何中断更高的优先级，则中断控制器发信号通知适当的处理器运行相应的中断服务例程810并将中断服务寄存器中的相应位清零812。在该处理器完成中断服务例程的执行814后，该处理器通常可通过对I/O端口或存储器映射的寄存器进行读写来向中断设备指示该中断已被处理816。

[0073] 因此，对于自动EOI中断请求而言，无需处理器发出显式EOI命令。因为当中断被发送到该处理器时中断服务寄存器中的相应位已被清零812，所以自动EOI中断将不会阻塞其他中断的传递。因为中断在被传递时已被有效地EOI，所以不使用处理EOI命令通常所需的计算周期。

[0074] 对虚拟化的中断控制器的使用允许VMM在物理中断控制器不支持选择性地EOI物理中断的功能时执行这一功能。这可通过维护待决EOI(即，需要在稍后被EOI的中断)的列表来实现。例如，当来宾操作系统发出EOI命令时，VMM可检查正被EOI的中断是否的确是物理中断控制器中的最高优先级的运行中中断。如果不是，则VMM将该中断添加到待决

EOI 的列表即可。另一方面，如果正被 EOI 的中断是最高优先级的运行中中断，则 VMM 不仅 EOI 当前中断，而且 EOI 待决 EOI 的列表上的其他中断。

[0075] 分区间消息传送

[0076] 某些 VMM 将中断用作分区间消息传送的基础。分区是由 VMM 强制实施的隔离边界并且是虚拟机的“容器”。如果在一个分区中运行的软件需要与在同一机器上的第二分区中运行的软件进行通信，则它能够通过使用分区间消息来这样做。这些消息通常包含少量的净荷。例如，在一个已知管理程序的情况下，其消息净荷可包括多达 240 字节加上 16 字节的首部。当消息被发送时，它由管理程序来排队直到与目的地分区相关联的虚拟处理器准备执行。此时，管理程序可将中断传递给该虚拟处理器。这导致调用相应的 ISR。ISR 负责读取消息并对其内容作出反应。如上所述，ISR 一般必须在中断已被服务后“EOI”该中断。在这种情况下，EOI 将在消息被读取之后发送。分区间消息通信必须尽可能快。传统 EOI 机制增添了不合乎需要的开销，这表现在使用可能数万个周期来通知虚拟化的中断控制器消息已被读取并且可传递后续消息和较低优先级的待决中断。

[0077] 根据此处的本发明，用于发信号通知分区间消息已被处理的传统 EOI 机制的开销是可以避免的。在一个实施例中，为每一个 SINT 提供消息槽并且消息的布局由以下表 1 中所描述的数据结构来定义。

[0078] typedef struct

[0079] {

[0080] UINT8 Message Pending :1 ;

[0081] UINT8 Reserved :7 ;

[0082] }HV_MESSAGE_FLAGS ;

[0083] typedef struct

[0084] {

[0085] HV_MESSAGE_TYPE MessageType(消息类型) ;

[0086] UINT8 PayloadSize(净荷大小) ;

[0087] HV_MESSAGE_FLAGS MessageFlags(消息标志)

[0088] UINT8 Reserved[2] ;

[0089] union

[0090] {

[0091] HV_PARTITION_ID Sender(发送者) ;

[0092] HV_PORT_ID Port(端口) ;

[0093] } ;

[0094] }HV_MESSAGE_HEADER ;

[0095] # 定义 HV_MESSAGE_MAX_PAYLOAD_BYTE_COUNT(HV 消息最大净荷

[0096] 字节数)240

[0097] # 定义 HV_MESSAGE_MAX_PAYLOAD_QWORD_COUNT(HV 消息最大净

[0098] 荷 QWORD 数)30

[0099] typedef struct

[0100] {

[0101] HV_MESSAGE_HEADER Header(首部);

[0102] UINT64 Payload[HV_MESSAGE_MAX_PAYLOAD_QWORD_COUNT];

[0103] }HV_MESSAGE;

[0104] 表1

[0105] 图9是描绘根据此处的本发明的处理器间消息处理的实施例的流程图。发送处理器发送对应于已被指定为自动EOI的指定SINT的处理器间消息902。VMM将该消息附加到消息队列904并确定对应于该指定SINT的消息槽是否为空906。如果前一消息仍然存在于该消息槽中,则VMM在该槽中的消息的首部中置位消息待决位。如果该消息槽为空,则VMM将该消息复制到该消息槽910并且将与指定的SINT相关联的中断发送到接收处理器912。当在该接收处理器上运行的来宾OS接收到与SINT相关联的中断时,其ISR从该相应消息槽中读取该消息并基于消息类型和净荷来执行动作914。当该消息的处理完成时,该ISR清除该消息类型916。例如,根据表1中所定义的数据结构,ISR可通过将一指定值写入HV_MESSAGE_TYPE来清除该消息类型。该ISR然后检查刚刚处理的消息的消息待决位918。如果该消息待决位未被置位,意味着没有其他消息对该消息槽排队,则ISR无需执行其他动作920。这应当是最经常出现的情况。具体而言,ISR无需发送EOI命令,由此避免了相当多的计算开销。

[0106] 如果消息待决位已被设置,则该ISR向虚拟化的中断控制器发送消息结束(“EOM”)命令922,告诉VMM它应当重新尝试排队消息的传递。EOM命令的计算成本大致与EOI命令相同,但EOM仅在其他消息对消息槽排队的罕见情况下才被发送。因此,处理处理器间消息的平均成本被显著地降低。

[0107] 尽管已结合各附图所示的各实施例描述了本发明,但可以理解,可以使用类似的方面,或可以对所公开的各实施例的所述方面进行修改或添加来执行本发明的相同功能而不背离本发明。例如,在本发明的各方面,公开了用于提高虚拟化环境中的中断处理的操作效率的各种机制。然而,本文的教导还考虑了与这些描述方面等价的其它机制。因此,本发明应当不限于任何单一方面,而应按照所附权利要求书的宽度与范围来解释。

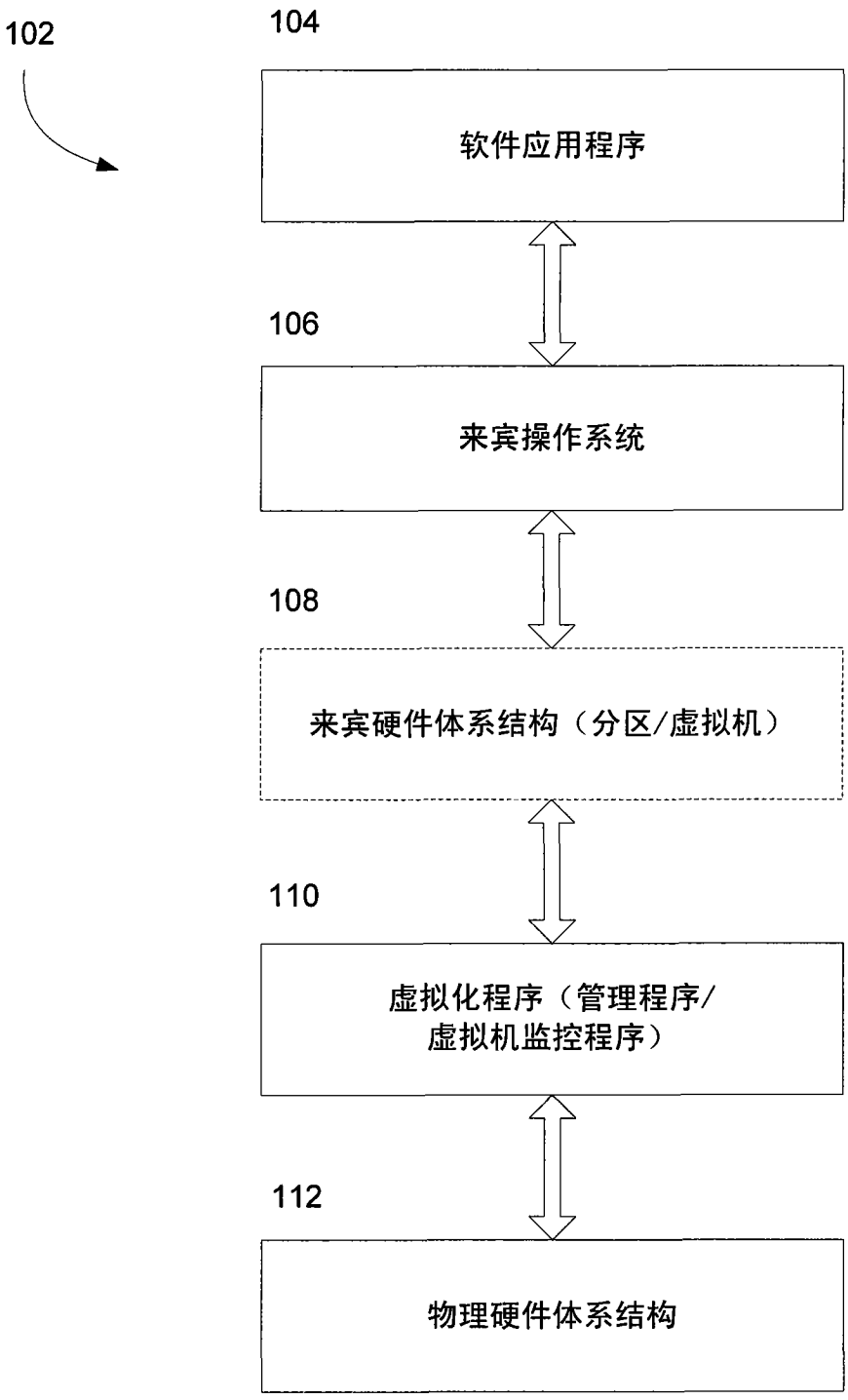

102
104
软件应用程序
106
来宾操作系统
108
来宾硬件体系结构（分区/虚拟机）
110
虚拟化程序（管理程序/
虚拟机监控程序）
112
物理硬件体系结构

图 1

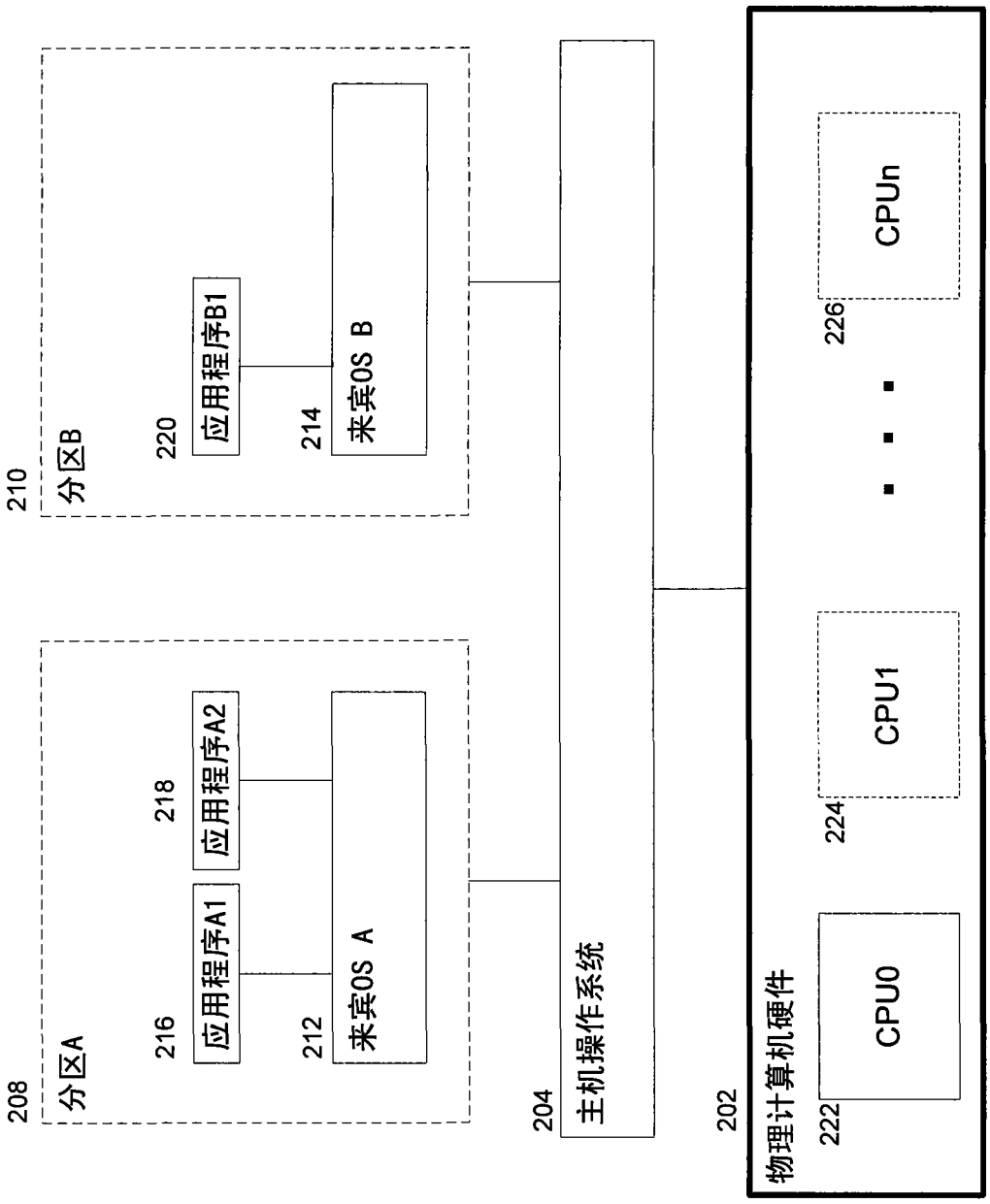

208
分区A
216
应用程序A1
218
应用程序A2
212
来宾OS A
210
分区B
220
应用程序B1
214
来宾OS B
204
主机操作系统
202
物理计算机硬件
222
CPU0
224
CPU1
226
CPUn

图 2

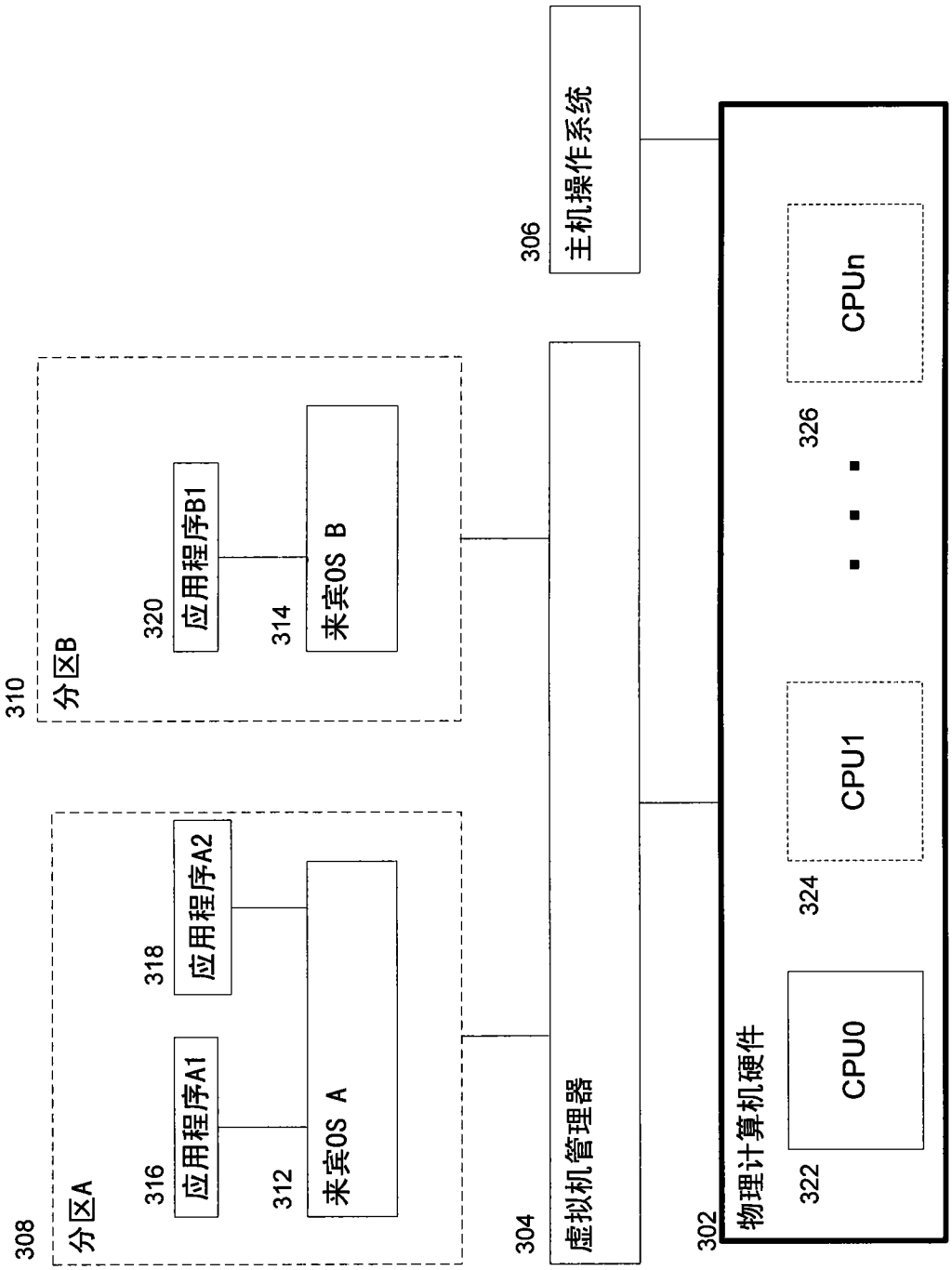

306
主机操作系统
310
分区B
320
应用程序B1
314
来宾OS B
308
分区A
316
应用程序A1
318
应用程序A2
312
来宾OS A
304
虚拟机管理器
302
物理计算机硬件
322
CPU0
324
CPU1
326
CPUn

图 3

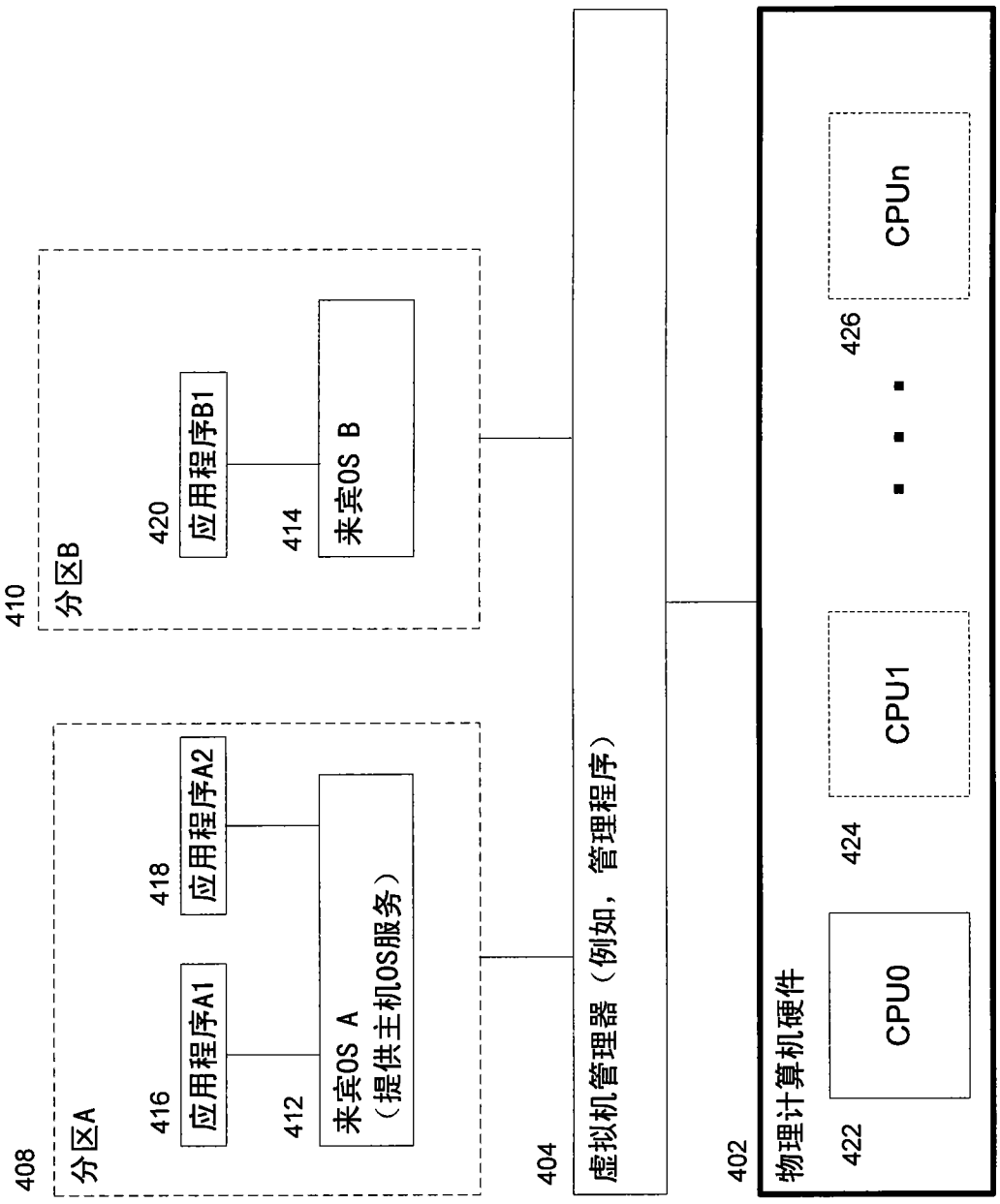

408
分区A
416
应用程序A1
418
应用程序A2
412
来宾OS A
（提供主机OS服务）
410
分区B
420
应用程序B1
414
来宾OS B
404
虚拟机管理器（例如，管理程序）
402
物理计算机硬件
422
CPU0
424
CPU1
426
CPUn

图 4

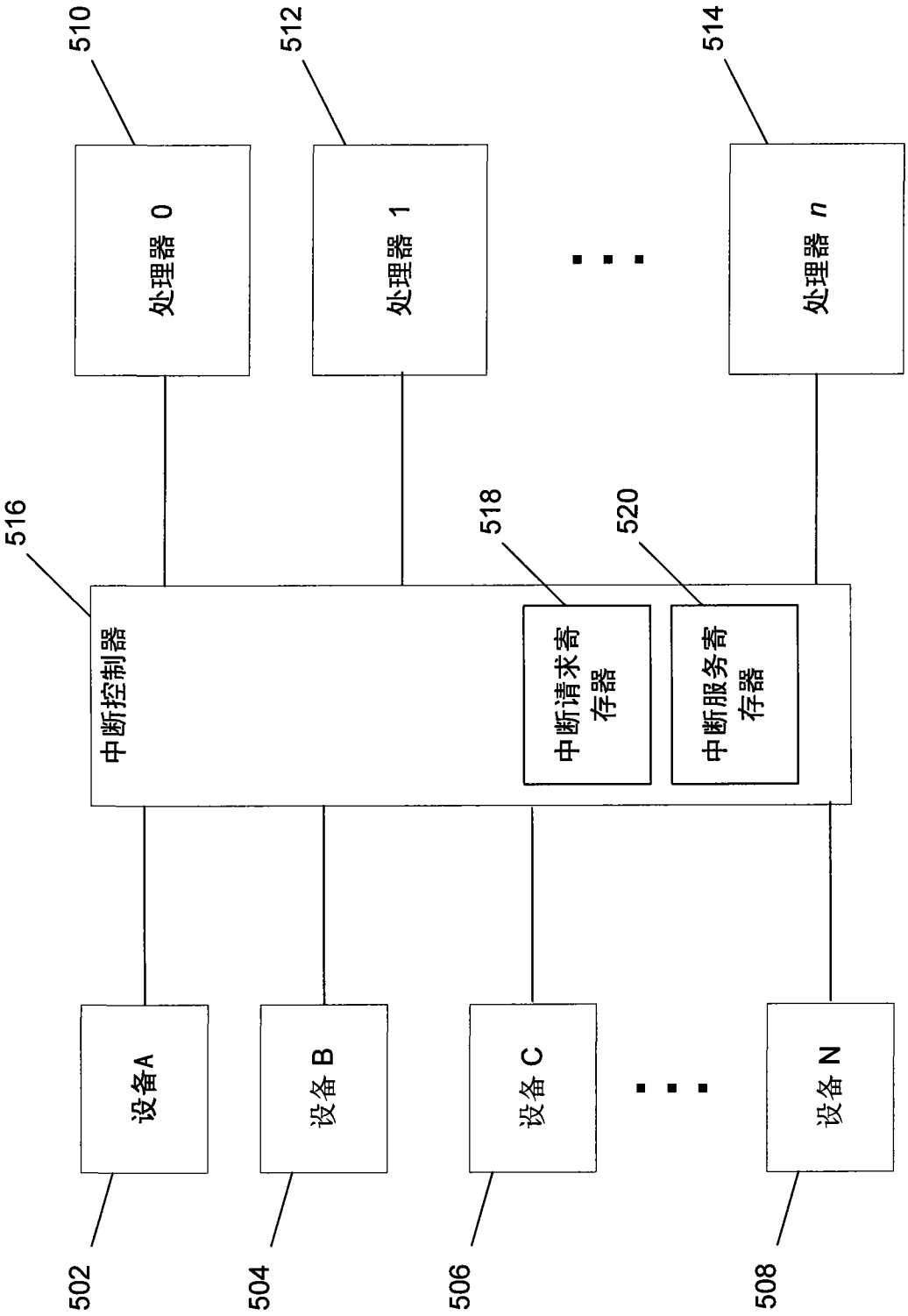

510
512
514
处理器 0
处理器 1
处理器 n
516
518
520
中断控制器
中断请求寄存器
中断服务寄存器
设备A
设备 B
设备 C
设备 N
502
504
506
508

图 5

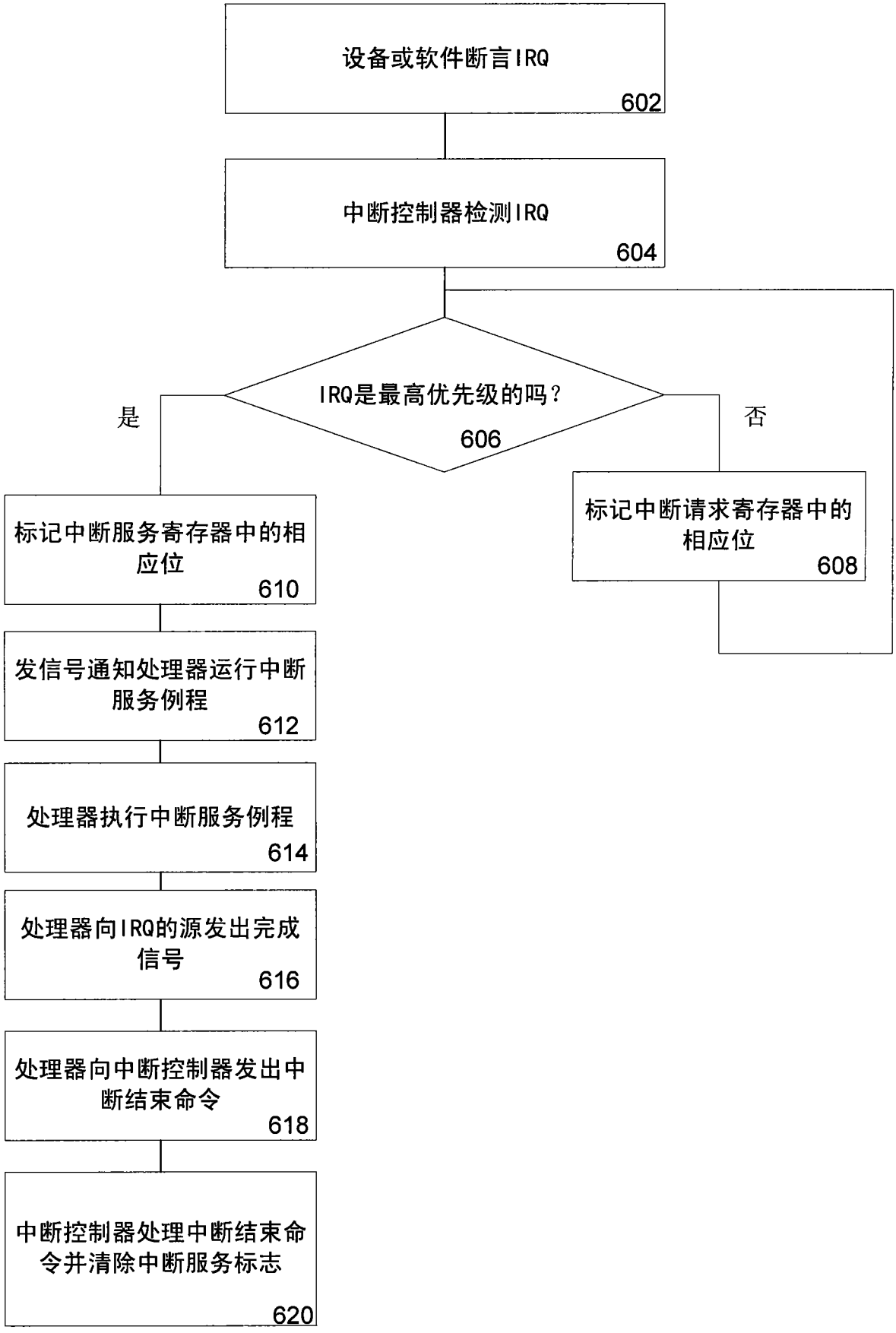

设备或软件断言IRQ
602
中断控制器检测IRQ
604
IRQ是最高优先级的吗？
606
是
否
标记中断服务寄存器中的相应位
610
发信号通知处理器运行中断服务例程
612
处理器执行中断服务例程
614
处理器向IRQ的源发出完成信号
616
处理器向中断控制器发出中断结束命令
618
中断控制器处理中断结束命令并清除中断服务标志
620
标记中断请求寄存器中的相应位
608

图 6

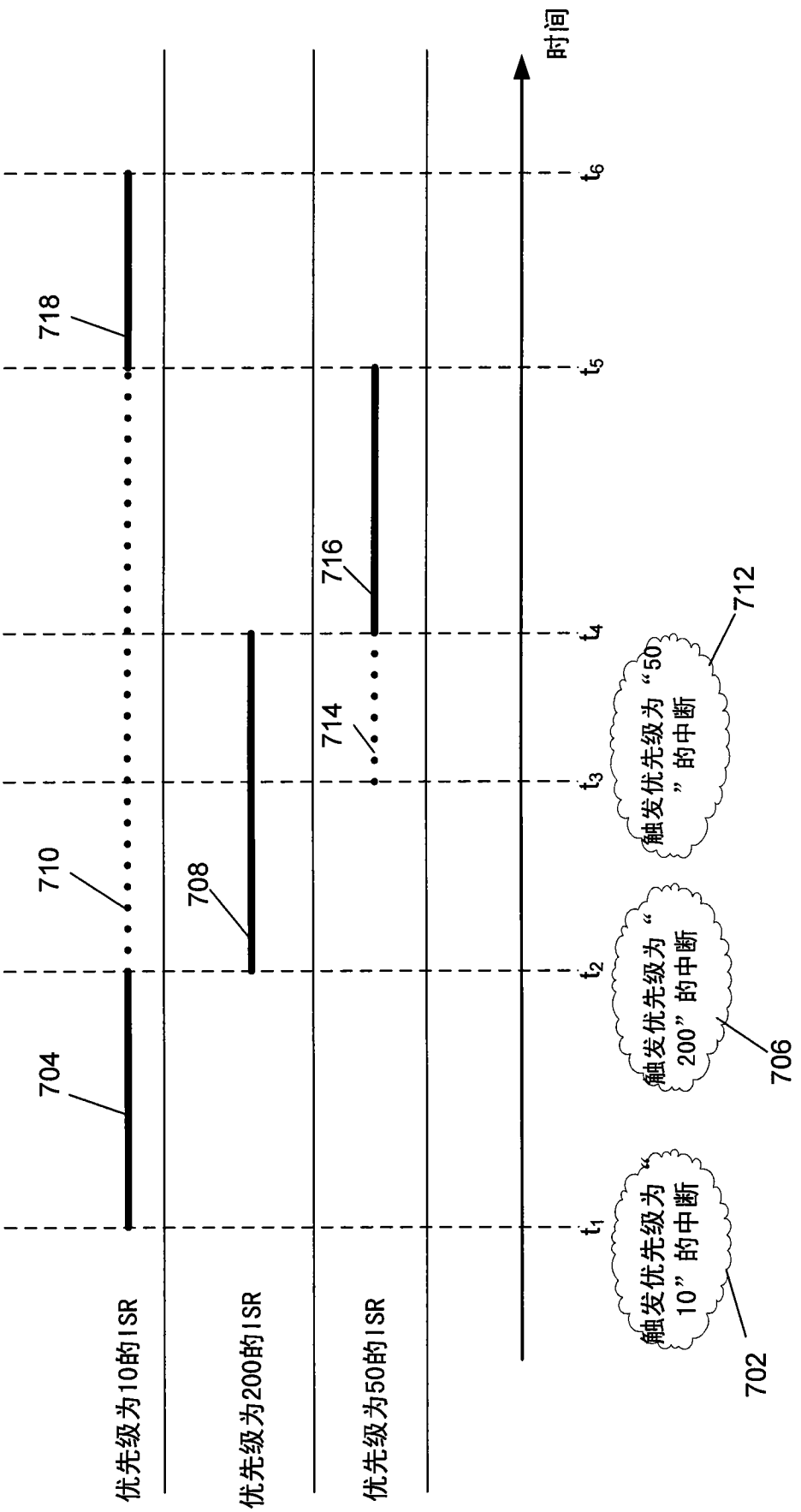

时间
t1
t2
t3
t4
t5
t6
优先级为10的ISR
优先级为200的ISR
优先级为50的ISR
704
708
710
714
716
718
触发优先级为“10”的中断
触发优先级为“200”的中断
触发优先级为“50”的中断
702
706
712

图 7

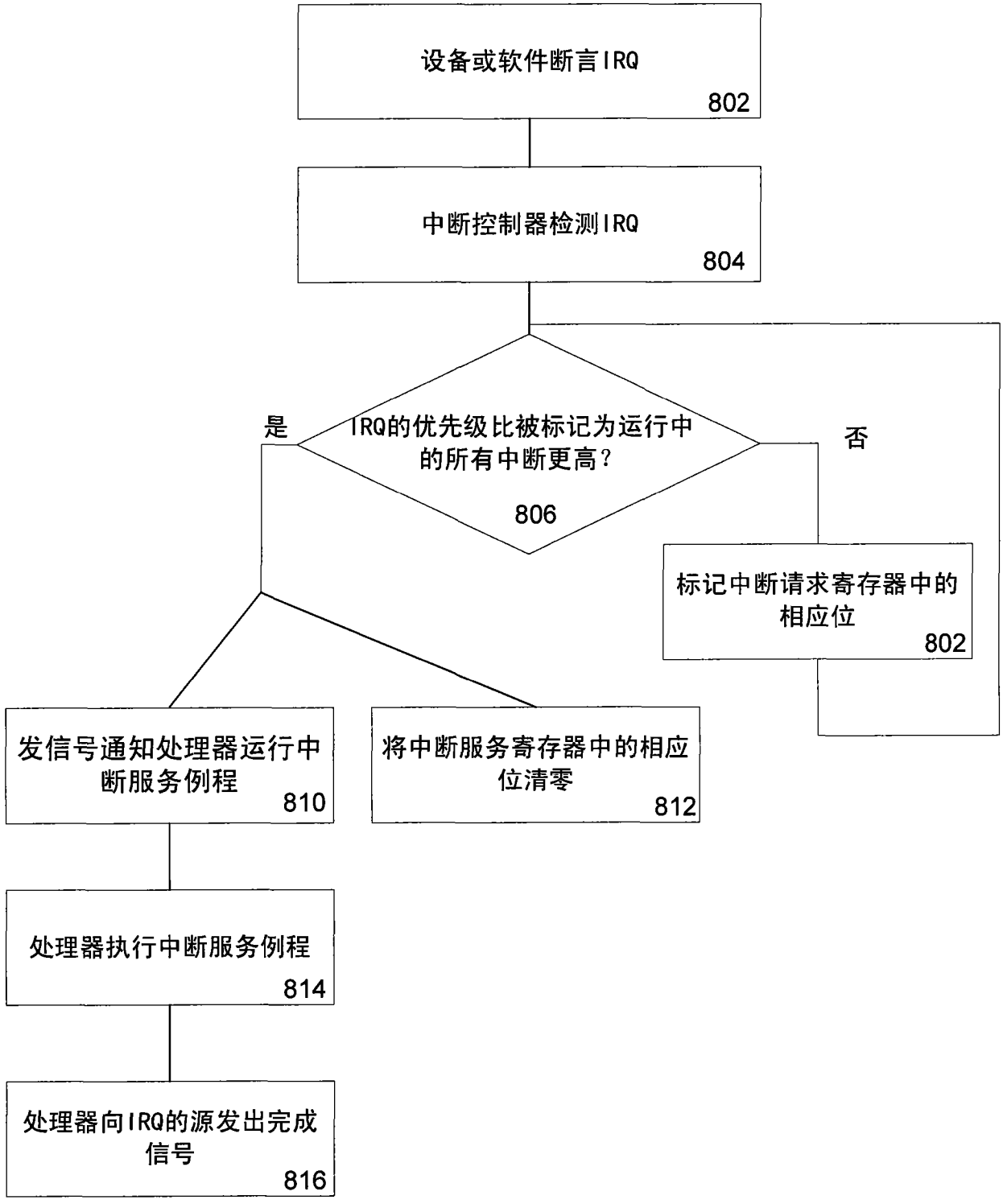

设备或软件断言IRQ
802
中断控制器检测IRQ
804
IRQ的优先级比被标记为运行中的所有中断更高？
806
是
否
标记中断请求寄存器中的相应位
802
发信号通知处理器运行中断服务例程
810
将中断服务寄存器中的相应位清零
812
处理器执行中断服务例程
814
处理器向IRQ的源发出完成信号
816

图 8

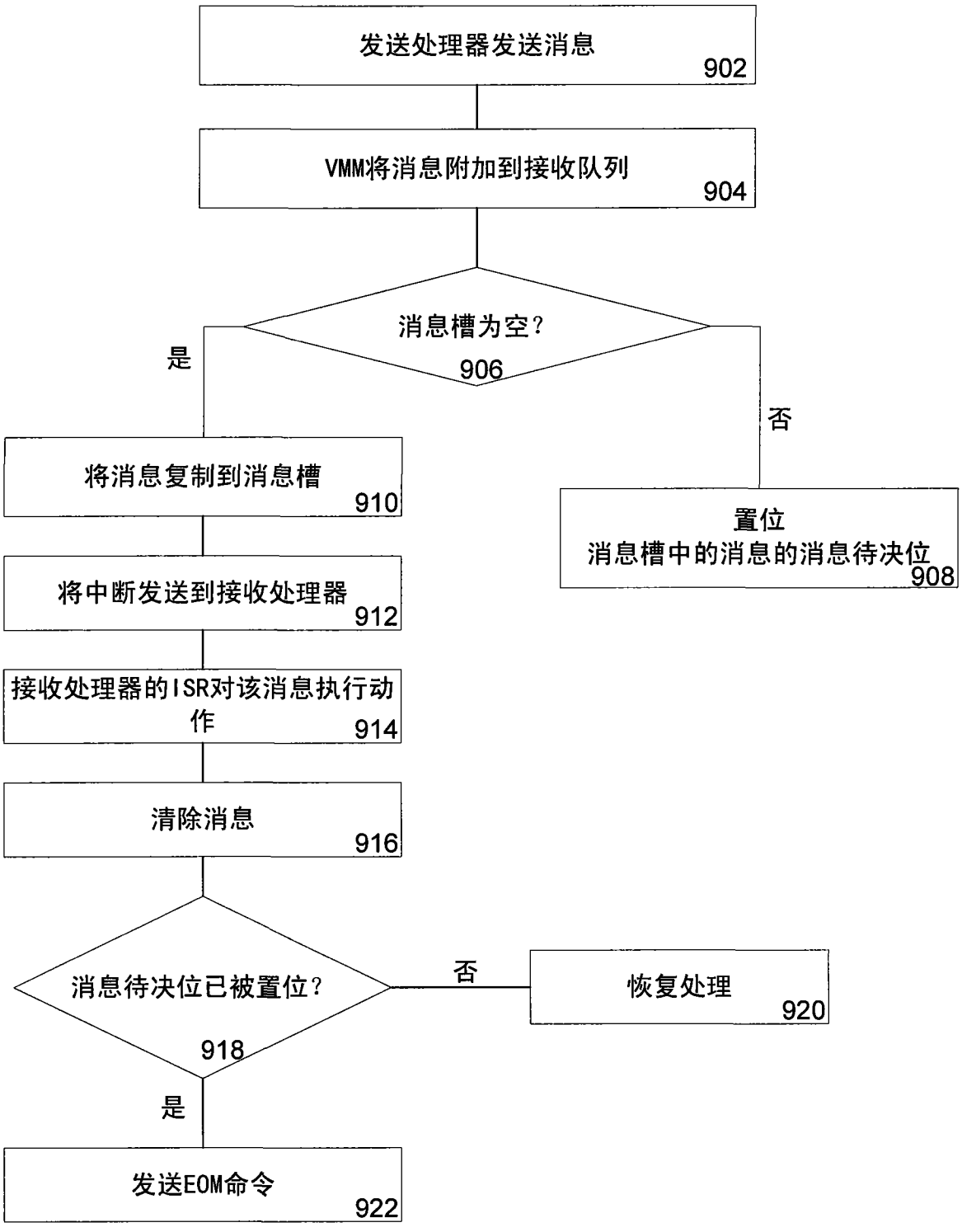

发送处理器发送消息
902
VMM将消息附加到接收队列
904
消息槽为空？
906
是
否
将消息复制到消息槽
910
置位
消息槽中的消息的消息待决位
908
将中断发送到接收处理器
912
接收处理器的ISR对该消息执行动作
914
清除消息
916
消息待决位已被置位？
918
否
恢复处理
920
是
发送EOM命令
922

图 9