(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 106845249 B
(45)授权公告日 2020.12.22

(21)申请号 201710096210.9
(22)申请日 2003.03.20
(65)同一申请的已公布的文献号
申请公布号 CN 106845249 A
(43)申请公布日 2017.06.13
(30)优先权数据
10/112,169 2002.03.29 US
(62)分案原申请数据
03811454.2 2003.03.20
(73)专利权人 英特尔公司
地址 美国加利福尼亚州
(72)发明人 詹姆斯·萨顿二世
戴维·格劳罗克
(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 黄嵩泉
(51)Int.Cl.
G06F 21/57(2013.01)
(56)对比文件
WO 0175565 A2,2001.10.11
WO 0116772 A1,2001.03.08
审查员 于萍

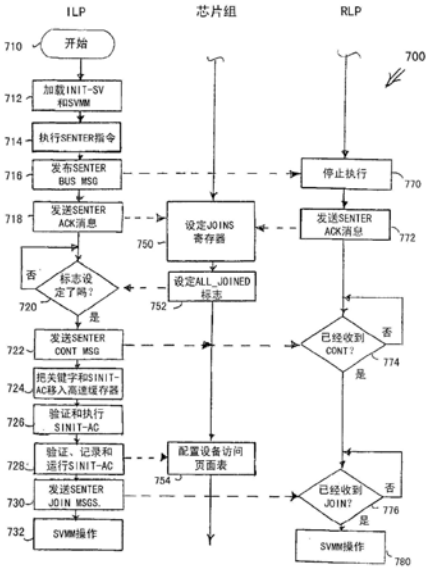
权利要求书4页 说明书10页 附图8页

(54)发明名称

用于执行安全环境初始化指令的系统和方法

(57)摘要

本申请公开了用于执行安全环境初始化指令的系统和方法。描述了在微处理器系统内启动安全操作的方法和装置。在一个实施方案中，一个启动逻辑处理器通过停止其它逻辑处理器的执行，然后把初始化和安全虚拟机监控软件载入存储器，来启动该过程。初始化处理器然后把初始化软件载入安全存储器进行验证和执行。初始化软件然后在安全系统操作之前验证和登记安全虚拟机监控软件。



CN 106845249 B

1.一种计算系统,包括:

存储设备,用于存储安全初始化程序代码和第二软件组件,所述安全初始化程序代码用于初始化安全处理环境,所述安全初始化程序代码的至少一部分将被复制到可信存储器中,所述第二软件组件用于防止来自不可信操作系统的对所述安全处理环境中的硬件资源的直接访问;以及

处理器,耦合至所述存储设备,所述处理器用于建立可用于确保后续操作能被信任的信任根,其中所述处理器用于执行所述安全初始化程序代码以初始化安全处理环境并且认证所述第二软件组件,并且其中所述处理器用于在所述第二软件组件被所述安全初始化程序代码认证之后并且所述信任根已经被建立之后执行所述第二软件组件。

2.如权利要求1所述的计算系统,还包括耦合至所述处理器的存储器设备。

3.如权利要求1所述的计算系统,还包括耦合至所述处理器的通用串行总线USB接口。

4.如权利要求1所述的计算系统,还包括耦合至所述处理器的可移动媒介接口。

5.一种处理器,包括:

所述处理器的可信存储器,用于支持安全初始化操作,

所述处理器的执行逻辑,所述执行逻辑包括安全初始化逻辑,所述安全初始化逻辑用于为安全执行环境建立信任根,

所述执行逻辑用于在所述安全执行环境内执行安全初始化程序代码,所述安全初始化程序代码的至少一部分将被复制到所述可信存储器中,

所述执行逻辑用于在执行安全初始化程序代码之前验证所述安全初始化程序代码,其中对所述安全初始化程序代码的验证基于散列算法,

所述执行逻辑用于验证虚拟机监控程序VMM,并且将VMM复制到所述安全执行环境内的可信系统存储区域。

6.如权利要求5所述的处理器,其特征在于,所述执行逻辑对所述VMM的验证包括对所述VMM的加密认证。

7.一种用于安全初始化的系统,包括:

多个处理器;

互连,用于耦合两个或更多个所述处理器;

系统存储器,耦合到所述两个或更多个处理器;

以及到至少一个存储设备的接口;

所述处理器中的至少一个处理器包括:

所述处理器的可信存储器,用于支持安全初始化操作,

所述处理器的安全初始化逻辑,用于为安全执行环境建立信任根,

所述处理器的执行逻辑,用于在所述安全执行环境内执行安全初始化程序代码,所述安全初始化程序代码的至少一部分将被复制到所述可信存储器中,

所述执行逻辑用于在执行安全初始化程序代码之前验证所述安全初始化程序代码,

所述执行逻辑用于验证虚拟机监控程序VMM,并且将VMM复制到所述安全执行环境内的可信系统存储区域。

8.如权利要求7所述的系统,还包括访问逻辑,用于许可或拒绝对于所述系统存储器中的存储器页面的访问。

9.如权利要求7所述的系统,其特征在于,所述执行逻辑对所述VMM的验证包括对所述VMM的加密认证。

10.如权利要求7所述的系统,还包括耦合到所述接口的存储设备。

11.如权利要求7所述的系统,其中所述接口是到外设部件互连PCI总线的接口。

12.如权利要求7所述的系统,其中所述接口是串行接口。

13.如权利要求7所述的系统,其中所述接口是到通用串行总线USB的接口。

14.如权利要求7所述的系统,其中所述接口是到低引线数LPC总线的接口。

15.如权利要求7所述的系统,其中所述接口是到集成驱动器电路IDE总线的接口。

16.如权利要求7所述的系统,其中所述接口是到小型计算机系统互连SCSI总线的接口。

17.一种用于安全初始化的系统,包括:

多个处理器;

互连,用于耦合两个或更多个所述处理器;

系统存储器,耦合到所述两个或更多个处理器;

以及到至少一个存储设备的接口;

所述处理器中的至少一个处理器包括:

所述处理器的可信存储器,用于支持安全初始化操作,

所述处理器的执行逻辑,所述执行逻辑包括安全初始化逻辑,所述安全初始化逻辑用于为安全执行环境建立信任根,

所述执行逻辑用于在所述安全执行环境内执行安全初始化程序代码,所述安全初始化程序代码的至少一部分将被复制到所述可信存储器中,

所述执行逻辑用于在执行安全初始化程序代码之前验证所述安全初始化程序代码,

所述执行逻辑用于验证虚拟机监控程序VMM,并且将VMM复制到所述安全执行环境内的可信系统存储区域。

18.如权利要求17所述的系统,还包括访问逻辑,用于许可或拒绝对于所述系统存储器中的存储器页面的访问。

19.如权利要求17所述的系统,其特征在于,所述执行逻辑对所述VMM的验证包括对所述VMM的加密认证。

20.一种用于安全初始化的系统,包括:

多个处理器;

互连,用于耦合两个或更多个所述处理器;

系统存储器,耦合到所述两个或更多个处理器;

以及到至少一个存储设备的串行接口;

所述处理器中的至少一个处理器包括:

所述处理器的可信存储器,用于支持安全初始化操作,

所述处理器的安全初始化逻辑,用于为安全执行环境建立信任根,

所述处理器的执行逻辑,用于在所述安全执行环境内执行安全初始化程序代码,所述安全初始化程序代码的至少一部分将被复制到所述可信存储器中,

所述执行逻辑用于在执行安全初始化程序代码之前验证所述安全初始化程序代码,

所述执行逻辑用于验证虚拟机监控程序VMM,并且将VMM复制到所述安全执行环境内的可信系统存储区域。

21.如权利要求20所述的系统,还包括访问逻辑,用于许可或拒绝对于所述系统存储器中的存储器页面的访问。

22.如权利要求20所述的系统,其特征在于,所述执行逻辑对所述VMM的验证包括对所述VMM的加密认证。

23.如权利要求20所述的系统,其中所述串行接口是到外设部件互连PCI总线的接口。

24.如权利要求20所述的系统,其中所述串行接口是到通用串行总线USB的接口。

25.一种芯片上系统SoC,包括:

处理器,该处理器包括:

所述处理器的可信存储器,用于支持安全初始化操作,

所述处理器的安全初始化逻辑,用于为安全执行环境建立信任根,

所述处理器的执行逻辑,用于在所述安全执行环境内执行安全初始化程序代码,所述安全初始化程序代码的至少一部分将被复制到所述可信存储器中,

所述执行逻辑用于在执行安全初始化程序代码之前验证所述安全初始化程序代码,

所述执行逻辑用于验证虚拟机监控程序VMM,并且将VMM复制到所述安全执行环境内的可信系统存储区域,以及

其中所述执行逻辑包括一个或多个核以用于执行一个或多个线程;

所述SoC还包括存储器控制器,用于将所述处理器耦合到系统存储器;

所述处理器和所述存储器控制器在单个半导体管芯上。

26.如权利要求25所述的芯片上系统SoC,还包括访问逻辑,用于许可或拒绝对于所述系统存储器中的存储器页面的访问,所述访问逻辑在所述单个半导体管芯上。

27.如权利要求25所述的芯片上系统SoC,其特征在于,所述执行逻辑对所述VMM的验证包括对所述VMM的加密认证。

28.一种芯片上系统SoC,包括:

处理器,该处理器包括:

所述处理器的可信存储器,用于支持安全初始化操作,

所述处理器的执行逻辑,所述执行逻辑包括安全初始化逻辑,所述安全初始化逻辑用于为安全执行环境建立信任根,

所述执行逻辑用于在所述安全执行环境内执行安全初始化程序代码,所述安全初始化程序代码的至少一部分将被复制到所述可信存储器中,

所述执行逻辑用于在执行安全初始化程序代码之前验证所述安全初始化程序代码,

所述执行逻辑用于验证虚拟机监控程序VMM,并且将VMM复制到所述安全执行环境内的可信系统存储区域;并且

其中所述执行逻辑包括一个或多个核以用于执行一个或多个线程;

所述SoC还包括存储器控制器,用于将所述处理器耦合到系统存储器;

所述处理器和所述存储器控制器在单个半导体管芯上。

29.如权利要求28所述的芯片上系统SoC,还包括访问逻辑,用于许可或拒绝对于所述系统存储器中的存储器页面的访问,所述访问逻辑在所述单个半导体管芯上。

30.如权利要求28所述的芯片上系统SoC,其特征在于,所述执行逻辑对所述VMM的验证包括对所述VMM的加密认证。

用于执行安全环境初始化指令的系统和方法

[0001] 本申请是针对分案申请201110030876.7再次提出的分案申请。分案申请201110030876.7是PCT国际申请号为PCT/US03/08762、国际申请日为2003年3月20日、进入中国国家阶段的申请号为03811454.2,题为“用于执行安全环境起始指令的系统和方法”的申请的分案申请。

技术领域

[0002] 本发明通常涉及微处理器系统,更具体地说,涉及可以在可信或安全环境中运行的微处理器系统。

背景技术

[0003] 在本地或远程微型计算机上执行的金融和个人事务的增加量已经推动了“可信”或“安全”微处理器环境的建立。这些环境试图要解决的问题是个人隐私的泄露或者数据被破坏或滥用。用户不想公开他们的私人数据。他们也不想不恰当的事务改变或使用他们的数据。这样的例子包括非故意地泄露医疗记录或者从在线银行或其它存款处的资金电子失窃。类似地,内容供给者设法保护数字内容(例如,音乐、其它音频、视频或其它类型的一般数据)不会在未经授权的情况下被复制。

[0004] 现有的可信系统可以利用一套完全封闭的可信软件。这个方法实施起来相对简单,但缺点是不允许同时使用市场上可买到的普通操作系统和应用软件。这个缺点限制了对上述可信系统的认可程度。

附图说明

[0005] 本发明是以实施例的方式来说明的,而不是以限定的方式来说明的,附图中相近的附图标记表示类似的部件,其中:

[0006] 图1是在微处理器系统中执行的示例性软件环境的图。

[0007] 图2是依据本发明一个实施方案的某些示例性可信或安全软件模块和示例性系统环境的图。

[0008] 图3是依据本发明一个实施方案的示例性可信或安全软件环境的图。

[0009] 图4A是依据本发明一个实施方案,适合于支持图3的安全软件环境的示例性微处理器系统示意图。

[0010] 图4B是依据本发明另一实施方案,适合于支持图3安全软件环境的示例性微处理器系统示意图。

[0011] 图5是依据本发明另一实施方案,适合于支持图3安全软件环境的示例性微处理器系统示意图。

[0012] 图6是依据本发明一个实施方案的软件组件的执行的时线图。

[0013] 图7是依据本发明一个实施方案的软件和其它过程块的流程图。

具体实施方式

[0014] 下面的说明描述了在微处理器系统内初始化可信或安全环境的技术。在下面说明中，为了更彻底地理解本发明，阐述了许多具体细节，例如逻辑实现、软件模块分配、加密技术、总线信令技术，以及操作细节。然而，本领域技术人员将能理解，没有上述具体细节也可以实施本发明。在其它情况下，为了不混淆本发明，没有详细表示控制结构、门电平电路和全部软件指令序列。获悉本文所包含的说明的本领域普通技术人员无需超出常规的试验就能够实现恰当的功能性。本发明是以微处理器系统的形式公开的。然而，以其它处理器的形式也可以实施本发明，例如数字信号处理器、小型计算机或大型计算机。

[0015] 现在参考图1，图中所示为在微处理器系统中执行的一个示例性软件环境。图1中所示的软件不是可信的(非可信的)。当在高特权级下运行时，操作系统150的大小和持续更新使得按照适时方式进行任何信任分析是非常困难的。许多操作系统位于特权环(ring)零(0)内，即最高特权级。应用152，154和156具有降低了很多的特权，典型地位于特权环三(3)内。基于作出的决策以信任由操作系统150提供的设备，不同特权环的存在以及将操作系统150和应用152，154，156分开为这些不同特权环似乎允许图1的软件按照可信模式运行。然而，实际上进行上述信任决策经常是不切实际的。影响这个问题的因素包括运行系统150的大小(代码行的数目)，操作系统150可以是许多更新(新代码模块和补丁)的接收者的事实，以及操作系统150也可以包含由用户而不是操作系统开发者提供的代码模块(例如设备驱动器)的事实。操作系统150可以是通用操作系统，例如Microsoft®Windows®，Linux或Solaris®，或者可以是任何其它适当已知或另外可获得的操作系统。应用或操作系统运行或正在运行的具体类型或名称不是关键的。

[0016] 现在参考图2，图中所示为依据本发明一个实施方案的某些示例性可信或安全软件模块和示例性系统环境200。在图2的实施方案中，处理器202、处理器212、处理器222和可选的其它处理器(未图示)被图示为单独硬件实体。在其它实施方案中，正如不同部件和功能单元的边界可以变化，处理器的数量也可以不同。在某些实施方案中，可以用在一个或多个物理处理器上运行的单独硬件执行线程(thread)或“逻辑处理器”来替换这些处理器。

[0017] 处理器202，212，222可以包含某些专用电路或逻辑元件以支持安全或可信操作。例如，处理器202可以包含安全输入(SENTER)逻辑204以支持执行专用SENTER指令，所述SENTER指令可以初始化可信操作。处理器202也可以包含总线消息逻辑206以支持系统总线230上的支持专用SENTER操作的专用总线消息。在另外的实施方案中，芯片组240的存储控制功能可以分配给处理器内的电路，对于多个处理器而言，可以包括在单个管芯上。在这些实施方案中，专用总线消息也可以在这些处理器内部的总线上发送。由于几个原因，使用专用总线消息可以增加系统安全性或可信任性。如果电路元件例如处理器202，212，222或芯片组240包含本发明公开的实施方案的适当逻辑元件，则它们可以只发布或响应上述消息。因此专用总线消息的成功交换可以有助于确保适当的系统配置。专用总线消息也可以允许通常应该被禁止的活动，例如复位平台配置寄存器278。通过允许专用总线消息只响应于专用安全指令而发布，可以限制潜在的不友善的非可信代码对某些总线事务进行侦测的能力。

[0018] 另外，处理器202可以包含安全存储器208以支持安全初始化操作。在一个实施方案中，安全存储器208可以是处理器202的内部高速缓存器，或许按照专用模式运行。在另外

的实施方案中，安全存储器208可以是专用存储器。其它处理器，例如处理器212和处理器222，也可以包括SENTER逻辑214，224、总线消息逻辑216，226、以及安全存储器218，228。

[0019] “芯片组”可被定义为一组电路和逻辑，它们支持针对连接的一个或多个处理器所进行存储器和输入/输出(I/O)操作。芯片组的诸多单独元件可以被集合在单个芯片上、在一对芯片上或分散在多个芯片中，包括处理器。在图2的实施方案中，芯片组240可以包括支持存储器和I/O操作的电路和逻辑，以支持处理器202，212和222。在一个实施方案中，芯片组240可以与许多存储页面250-262和设备访问页面表248进行交互，页面表248包含指示非处理器设备是否可以访问存储页面250-262的控制信息。芯片组240可以包括设备访问逻辑247，所述设备访问逻辑可以允许或拒绝从I/O设备到存储页面250-262的所选部分的直接存储器存取(DMA)。在某一实施方案中，设备访问逻辑247可以包含允许或拒绝上述访问所需要的所有相关信息。在其它实施方案中，设备访问逻辑247可以访问保存在设备访问页面表248内的上述信息。存储页面的实际数量不是重要的，并且将根据系统需求而变化。在其它实施方案中，存储器访问功能可以在芯片组240的外部。在另外实施方案中，芯片组240的功能还可以在一个或多个物理设备中分配。

[0020] 为支持专用SENTER操作，芯片组240可以另外包括它自己的总线消息逻辑242来支持系统总线230上的专用总线消息。这些专用总线消息中的某些可以包括：把密钥(key)寄存器244的内容传递给处理器202，212或222，或者允许通过处理器202，212或222检验专用的ALL-JOINED标志274。总线消息逻辑242的附加特征可以是把多个处理器的总线活动登记在“EXISTS”寄存器272中以及把多个处理器的某一专用总线消息活动保存在“JOINS”寄存器272中。EXISTS寄存器272和JOINS寄存器272的内容的等同性可以用来设定专用的ALL-JOINED标志，以指示系统内所有处理器都在参与安全输入过程。

[0021] 芯片组240可以支持I/O总线上的标准I/O操作，所述I/O总线例如外设部件接口(PCI)、加速图形接口(AGP)、通用串行总线(USB)、低引线数(LPC)总线或任何其它类型I/O总线(未示出)。接口290可以用来使芯片组240与令牌276连接，令牌276包含一个或多个平台配置寄存器(PCR)278，279。在一个实施方案中，接口290可以是通过修改增加了某些安全上的增强的LPC总线(低引线数(LPC)接口规范，英特尔公司1997年12月29日的修订版1.0)。上述安全上的增强的一个实施例是位置确认消息，利用以前保存的消息头和地址信息，把令牌276内的平台配置寄存器(PCR)278作为目标。在一个实施方案中，令牌276可以包含专用安全特征，在一个实施方案中，可以包括可信平台模块(TPM)281，该模块在2001年12月1日由TCPA出版的版本为1.1a的可信计算平台联合(TCPA)主要规范中被公开(在本申请递交时从www.trustedpc.com可得到)。

[0022] 在系统环境200内确定的两个软件组件是安全虚拟机监控程序(SVMM)282模块和安全初始化授权码(SINIT-AC)280模块。SVMM 282模块可以被保存在系统盘或其它大容量存贮器上，并且根据需要被移动或复制到其它位置。在一个实施方案中，在开始安全启动过程之前，SVMM 282可以被移动或复制到一个或多个存储页面250-262。安全输入过程之后，可以创建虚拟机环境，其中：SVMM 282可以作为系统内最高特权代码来运行，并可以被用来允许或拒绝在已创建的虚拟机内的操作系统或应用直接访问某些系统资源。

[0023] 安全输入过程所需要的某些动作可能超出简单硬件实施的范围，并且相反可以有利地使用软件模块，其中所述软件模块的执行可以默认是可信的。在一个实施方案中，通过

安全初始化(SINIT)代码可以执行这些动作。这里确定三个典型动作,但这些动作不应理解为是限定性的。一个动作可能要求对各种表示系统配置关键部分的控制进行检验,以确保所述配置支持正确的安全环境实例化。在一个实施方案中,一个要求的检验可以是,芯片组240提供的存储控制器配置不允许两个或多个不同系统总线地址接触存储页面250-262内的相同位置。第二个动作可以是配置设备访问页面表248和设备访问逻辑247,以保护SVMM 282的存储驻留拷贝使用的那些存储页面不受非处理器设备干扰。第三个动作可以是计算和登记SVMM 282模块的身份,并且把系统控制传递给它。这里“登记(register)”是指把SVMM282的信任测量结果放入寄存器,例如放入PCR278或放入PCR279。当进行了这个最后的动作,潜在的系统用户可以检查SVMM282的可信度。

[0024] 处理器或芯片组的制造商可以生成SINIT代码。为此,可以信任SINIT代码来帮助芯片组240的安全启动。为了分配SINIT代码,在一个实施方案中,众所周知的加密散列由全部SINIT代码构成,生成一个被称为“摘要”的值。一个实施方案生成一个160位的值来作为摘要。然后通过在一个实施方案中由处理器制造商拥有的私钥(private key)对摘要进行加密,以形成数字签名。当SINIT代码与相应数字签名捆绑在一起时,这个组合可以称为SINIT授权码(SINIT-AC)280。如下所述,SINIT-AC 280的拷贝可以在后面被验证。

[0025] SINIT-AC 280可以被保存在系统盘或其它大容量存储器上或者被保存在固定媒介中,并且根据需要被移动或复制到其它位置。在一个实施方案中,在开始安全启动过程之前,SINIT-AC 280可以被移动或复制到存储页面250-262以形成SINIT-AC存储驻留拷贝。

[0026] 任何逻辑处理器可以开始安全启动过程,并且因而可以被称为初始化逻辑处理器(ILP)。在本实施例中,处理器202为ILP,尽管系统总线230上的任何处理器能够成为ILP。此时,SINIT-AC280的存储驻留拷贝或SVMM282的存储驻留拷贝都不被认为是可信的,因为除了其它原因之外,另外的处理器或DMA设备可以重写存储页面250-262。

[0027] 然后,ILP(处理器202)执行专用指令。这个专用指令可以称为安全输入(SENTER)指令,并且可以由SENTER逻辑204支持。SENTER指令的执行可以使ILP(处理器202)在系统总线230上发布专用总线消息,然后为随后的系统动作等待相当长的时间间隔。SENTER执行开始之后,这些专用总线消息之一,即SENTER BUS MESSAGE在系统总线230上广播。除了ILP之外的那些逻辑处理器可以称为响应逻辑处理器(RLP),它们用内部非屏蔽事件响应SENTER BUS MESSAGE。在本实施例中,RLP包括处理器212和处理器222。RLP必须各自终止当前操作,在系统总线230上发送RLP确认(ACK)专用总线消息,然后进入等待状态。应该注意,ILP也在系统总线230上发送它自己的ACK消息。

[0028] 芯片组240可以包含一对寄存器,即“EXISTS”寄存器270和“JOINS”寄存器272。这些寄存器可以被用来检验ILP和所有RLP正在适当地响应SENTER BUS MESSAGE。在一个实施方案中,通过在逻辑处理器所进行的任何系统总线事务中把“1”写入EXISTS寄存器270的相应位,芯片组240可以始终跟踪在系统内的所有操作逻辑处理器。在本实施方案中,系统总线230上的每个事务必须包含标识字段(field),所述标识字段包含逻辑处理器标识符。在一个实施方案中,这是由物理处理器标识符和每个物理处理器内硬件执行线程的标识符构成的。例如,如果在处理器222上执行的线程在系统总线230上引起任何总线事务,则芯片组240将在该事务中发现这个逻辑处理器标识符,并且把“1”写入EXISTS寄存器270内的相应位置286。安全启动过程期间,当处理器222上的那个同一线程在系统总线230上发送它自己

的ACK信息时，芯片组240也将发现此情况，并且把“1”写入JOINS寄存器272内的相应位置288。(在图2的实施例中，为了清楚起见，每个物理处理器被图示为只带有单个执行线程。在另外实施方案中，物理处理器可以支持多个线程，因而支持多个逻辑处理器。)当JOINS寄存器272的内容与EXISTS寄存器270的内容匹配时，则芯片组240可以设置ALL-JOINED标志246，该标志表示所有处理器已经适当地响应了SENTER BUS MESSAGE。

[0029] 在另一实施方案中，在ALL-JOINED标志246设置之后，EXISTS寄存器270和JOINS寄存器272可以继续有助于安全性。在ALL-JOINED标志246设置之后直到可信或安全操作结束期间，芯片组240可以继续监控并将总线周期与JOINS寄存器272相比较。在这期间，如果芯片组240在任何时候从不是在JOINS寄存器272内当前所确定的逻辑处理器中发现总线事务，则芯片组240可以假设这个逻辑处理器不知何故已经“出现”晚了。这将暗示上述逻辑处理器没有参加过安全启动过程，因此可能代表攻击者(安全威胁)。在这样的情况下，芯片组240可以适当地响应以把这个攻击者保持在安全环境之外。在一个实施方案中，芯片组240可以在这样的情况下强制系统复位。在第二个实施方案中，可通过每个逻辑处理器在ACK总线消息断言之后的每个事务中在系统总线上断言专用的保留信号，来实现类似的“晚到”处理器检测。在本实施方案中，在ALL-JOINED标志246设置之后，如果芯片组240观察到处理器所初始化的总线事务没有专用的断言信号，则芯片组240可以再次假设这个逻辑处理器不知何故已经“出现”晚了，并且可能代表攻击者。

[0030] 发布SENTER BUS MESSAGE之后，ILP(处理器202)轮询ALL-JOINED标志246以发现所有处理器何时且是否已经用它们的ACK适当地进行了响应。如果从未设置标志246，几种实现是可能的。在ILP或芯片组内、或其它地方的监控定时器可以使系统复位。可选地，系统可能中止并需要操作员复位。在任一情况下，尽管系统可能不继续运行，但安全环境断言得到保护(其中如果不是所有的处理器都参与，安全启动过程就不结束)。在正常操作中，在短时间之后，ALL-JOINED标志246被设置，并且ILP可以确保所有其它逻辑处理器已经进入等待状态。

[0031] 当ALL-JOINED标志246被设置时，为了验证和随后执行包含在SINIT-AC280内的SINIT代码，ILP(处理器202)可以把SINIT-AC 280的拷贝和密钥284都移入安全存储器208。在一个实施方案中，这个安全存储器208可以是ILP(处理器202)的内部高速缓存器，或许按照专用模式运行。密钥284表示与私钥对应的公钥(public key)，私钥被用来加密包含在SINIT-AC 280模块内的数字签名，并且密钥284被用来检验数字签名和由此验证SINIT代码。在一个实施方案中，密钥284可能已经被保存在处理器内，或许作为SENTER逻辑204的一部分。在另一实施方案中，密钥284可以被保存在芯片组240的只读密钥寄存器244内，只读密钥寄存器244由ILP读取。在又一实施方案中，不是处理器就是芯片组密钥寄存器244可以实际保存密钥284的加密摘要，其中密钥284本身被包含在SINIT-AC 280模块内。在最后这个实施方案中，ILP从密钥寄存器244中读取摘要，在嵌入在SINIT-AC 280内的密钥284之上计算等同加密散列(hash)，并且比较这两个摘要以确保所提供的密钥284是确实可信的。

[0032] 然后，SINIT-AC拷贝和公钥拷贝可以在安全存储器208内存在。通过使用公钥拷贝解密包含在SINIT-AC拷贝内的数字签名，ILP现在可以验证SINIT-AC拷贝。所述解密产生加密散列摘要的原始拷贝。如果新计算出的摘要与这个原始摘要匹配，则SINIT-AC拷贝和它包含的SINIT代码可以被认为是可信的。

[0033] ILP现在可以经由系统总线230发布另一专用总线消息，即SENTER CONTINUE MESSAGE，向等待的RLP(处理器212，处理器222)和芯片组240信令将要初始化安全操作。如下所概述的那样，通过把SINIT-AC模块的加密摘要值写入安全令牌276内的平台配置寄存器272中，ILP现在可以登记SINIT-AC模块的唯一身份。通过把执行控制传递给保存在ILP安全存储器208内的SINIT代码的可信拷贝，ILP对其SENTER指令的执行现在可以终止。可信SINIT代码然后可以执行它的系统测试和配置动作，并且依照上述“登记”的定义，可以登记SVMM存储驻留拷贝。

[0034] 可以按照几种方式完成SVMM存储驻留拷贝的登记。在一个实施方案中，运行在ILP上的SENTER指令把计算出的SINIT-AC摘要写入安全令牌276内的PCR 278中。随后，可信SINIT代码可以把计算出的存储驻留SVMM摘要写入同一PCR 278或安全令牌276内的另一PCR 279中。如果把SVMM摘要写入同一PCR 278中，则安全令牌276用新值(SVMM摘要)对原始内容(SINIT摘要)进行散列化操作，并且把结果写回PCR278。在对PCR278的第一次(初始化)写入被限制在SENTER指令的这些实施方案中，结果摘要可以被用作系统的信任根(root of trust)。

[0035] 一旦可信SINIT代码已经结束它的执行，并且已经把SVMM的身份登记在PCR内，SINIT代码就可以把ILP执行控制传递给SVMM。在典型的实施方案中，ILP执行的最初的SVMM指令可代表SVMM的自初始化例程。在一个实施方案中，ILP可以把单独的RLP JOIN MESSAGE专用总线消息发送给每个RLP，使每个RLP在现在执行的SVMM拷贝的监督下加入操作。根据前面这个观点，如下面图3的讨论中所概述的那样，整个系统运行在可信模式下。

[0036] 现在参考图3，图中所示为依据本发明一个实施方案的示例性可信或安全软件环境。在图3的实施方案中，可以同时加载可信或非可信软件，并且可以在单个计算机系统上同时执行。SVMM350可选择地允许或防止来自一个或多个非可信操作系统340和非可信应用310-330的对硬件资源380的直接访问。在上下文中，“非可信”不是必定意味着操作系统或应用正在行为不端，但是意味着相互作用的代码的大小和多样性使得无法可靠地断言软件正在按要求运行，并且不存在干扰它执行的病毒或其它外来代码。在典型的实施方案中，非可信代码是由在当今个人计算机上可以找到的普通操作系统和应用所组成的。

[0037] SVMM350也可选择地允许或防止来自一个或多个可信或安全核心程序360和一个或多个可信应用370的对硬件资源380的直接访问。可以限制上述可信或安全核心程序360和可信应用370的大小和功能性，从而有助于在其上面完成信任分析的能力。可信应用370可以是在安全环境中可执行的任何软件代码、程序、例程或例程组。因此，可信应用370可以是各种应用或代码序列，或者可以是相对小的应用，例如Java程序。

[0038] 由操作系统340或核心程序360正常执行的能改变系统资源保护或特权的指令或操作可以被SVMM350俘获，并且可选择地被允许、部分允许或拒绝。作为实施例，在典型的实施方案中，由操作系统340或核心程序360正常执行的更改处理器页面表的指令将被SVMM350俘获，这将确保所述请求不试图要求改变在它的虚拟机范围之外的页面特权。

[0039] 现在参考图4A，图中所示为适合于支持图3的安全软件环境的微处理器系统400的一个实施方案。CPU A 410、CPU B 414、CPU C 418和CPU D 422可被配置以附加微代码或逻辑电路以支持专门指令的执行。在一个实施方案中，这个附加微代码或逻辑电路可以是图2的SENTER逻辑204。这些专用指令可以支持专用总线消息在系统总线420上的发布，系统总

线420可以使这些处理器在启动安全环境期间能够适当同步。在一个实施方案中，专用总线消息的发布可以由例如图2的总线消息逻辑206之类的电路来支持。类似地，芯片组430可以类似于芯片组420，并且可以支持上述系统总线420上的专用周期。物理处理器的数量可以根据具体实施方案的实施而变化。在一个实施方案中，处理器可以是Intel®Pentium®级的微处理器。经由PCI总线446，或者可选择地，经由USB 442、集成控制器电路(IDE)总线(未图示)、小型计算机系统接口(SCSI)总线(未图示)、或任何其它I/O总线，芯片组430可以与大容量存储设备进行交互，例如固定媒介444或可移动媒介448。固定媒介444或可移动媒介448可以是磁盘、磁带、磁碟、磁光驱动器、CD-ROM、DVD-ROM、闪存卡，或许多其它形式的大容量存储器。

[0040] 在图4A的实施方案中，四个处理器CPU A 410、CPU B 414、CPU C 418和CPU D 422图示为四个单独硬件实体。在其它实施方案中，处理器的数量可以不同。实际上，物理上离散的处理器可以用在一个或多个物理处理器上运行的分立的硬件执行线程来替换。在后者的情况下，这些线程拥有许多附加物理处理器的特征。为了具有一般的表述来讨论使用多个物理处理器和多个在处理器上的线程的任何混合，表述“逻辑处理器”可以用来描述一个物理处理器或在一个或多个物理处理器内操作的线程。因此，一个单线程处理器可以被认为是一个逻辑处理器，且多线程或多核心处理器可以被认为是多个逻辑处理器。

[0041] 在一个实施方案中，芯片组430与改进的LPC总线450进行交互。改进的LPC总线450可以用来把芯片组430与安全令牌454连接。在一个实施方案中，令牌454可以包括由可信计算平台联合(TCPA)设想的TPM471。

[0042] 现在参考图4B，图中所示为适合于支持图3的安全软件环境的另一微处理器系统490的实施方案。与图4A的实施方案不同，CPU A 410和CPU B 414可使用系统总线A 402而连接到芯片组428，而CPU C 418和CPU D 422可使用系统总线B 404而连接到芯片组428。在其它实施方案中，可以使用两个以上的系统总线。在另一替代实施方案中，可以使用点对点总线。专用指令可以支持专用总线消息在系统总线A 402和系统总线B 404上的发布，专用总线消息在系统总线A 402和系统总线B 404上的发布可以使这些处理器在启动安全环境期间能够适当地同步。在一个实施方案中，专用总线消息的发布可以由例如图2的总线消息逻辑206之类的电路来支持。

[0043] 在一个实施方案中，芯片组428负责维护系统总线A 402和系统总线B 404上的一致性和相干性。如果在系统总线A 402上发送标准或专用的总线消息，芯片组428就会把这个信息(在适当时)反映在系统总线B 404上，反之亦然。

[0044] 在可替代的实施方案中，芯片组428把系统总线A 402和系统总线B 404看作独立子系统。在系统总线A 402上发布的任何专用总线消息只应用于该总线上的处理器，类似地，在系统总线B 404上发布的任何专用总线消息只应用于该总线上的处理器。针对系统总线A 402建立的任何受保护的存储器只可被连接到系统总线A 402的处理器所访问，而系统总线B 404上的处理器可以被看作非可信设备。为了获得对为系统总线A 402上的CPU A 410和CPU B 414建立的任何受保护的存储器的访问，系统总线B 404上的处理器CPU C 418和CPU D 422必须执行它们自己的SENTER过程，创建一个被登记在案的环境，该环境等同于为系统总线A 402上的处理器建立的环境。

[0045] 现在参考图5，表示依据本发明另一实施方案，适合于支持图3的安全软件环境的

示例性微处理器系统500的示意图。与图4A的实施方案不同，每个处理器(例如CPU A 510)可以包括某些芯片组功能(例如芯片组功能593)，所述芯片组功能执行例如存储控制器功能和设备访问逻辑功能。由此，这些芯片组功能允许存储器(例如存储器A 502)直接连接到处理器。其它芯片组功能可以保留在独立的芯片组530中。在系统总线520上可以发布专用总线消息。

[0046] 每个处理器可以间接访问连接到其它处理器的存储器，然而，与对处理器本身存储器的访问相比，这些访问可能相当慢。在开始SENTER过程之前，软件可以把SINIT-AC 566和SVMM 574的拷贝从固定媒介544移入本地存储器504，形成SINIT-AC 556拷贝和SVMM 572拷贝。在一个实施方案中，可以选择存储器504，因为它由确定为ILP的处理器直接访问，在图5实施例中，这是CPU B 514。可选择地，SINIT-AC 566和SVMM 574的拷贝可以放在连接到其它(非ILP)处理器的其它存储器中，只要ILP 514能够访问那些存储器。如图2中已经描述的那样，CPU B ILP 514通过发布SENTER指令开始安全输入过程，并且具有类似的结果和发布的总线周期。如上结合图2所述，芯片组530可以利用EXISTS寄存器576、JOINS寄存器580和ALL-JOINED标志584，以确定所有处理器是否已经适当地响应了SENTER BUS MESSAGE，并且把这个信息发送给ILP。ILP(CPU B 514)可以再次把SINIT-AC 556的存储驻留拷贝连同公钥564的拷贝一起移入安全存储器560。在SINIT-AC 556确认和登记后，ILP就可以继续进行SVMM 572的存储驻留拷贝的确认和登记。

[0047] 现在参考图6，表示依据本发明一个实施方案的各种操作的时线图。图6的时线表示结合图2所述的示例性系统所论述的全部操作的调度安排。当软件决定安全或可信操作是所期望的，在时间610，任何软件定位SINIT-AC 280和SVMM 282，并复制SINIT-AC 280和SVMM 282的拷贝以用于随后的SENTER指令。在本实施例中，软件把SINIT-AC 280拷贝和SVMM 282拷贝加载到一个或多个存储页面250-262中。然后选择一个处理器(在本实施例中是处理器202)作为ILP，ILP在时间612发布SENTER指令。在时间614，ILP的SENTER指令发布SENTER BUS MESSAGE 616。然后，在时间628进入“等待芯片组标志”状态之前，ILP在时间618发布它本身的SENTER ACK 608。

[0048] 每个RLP，例如处理器222，通过在时间620期间完成当前指令来响应SENTER BUS MESSAGE 616。然后，RLP发布它的SENTER ACK 622，然后进入状态634，其中它等待SENTER CONTINUE MESSAGE。

[0049] 芯片组240花费时间624来设定JOINS寄存器272以响应在系统总线230上观察到的SENTER ACK信息。当JOINS寄存器272的内容与EXISTS寄存器270的内容匹配时，芯片组240在时间626设定ALL-JOINED标志246。

[0050] 在此期间，ILP在轮询ALL-JOINED标志246时可以保持在循环状态下。当ALL-JOINED标志246被设定，并且ILP在时间630确定ALL-JOINED标志246被设定，然后，ILP可以在时间632期间发布SENTER CONTINUE MESSAGE。当SENTER CONTINUE MESSAGE在时间636在系统总线230上传播时，RLP可以进入“等待加入”的状态。例如，处理器222的RLP在时间周期638期间进入“等待加入”的状态。

[0051] 一旦发布SENTER CONTINUE MESSAGE，ILP于是(在时间周期640内)就可以把芯片组240的密钥寄存器244的公钥和SINIT-AC的拷贝加入它的安全存储器208，以形成所述公钥拷贝和SINIT-AC拷贝。在另一实施方案中，密钥寄存器244可以包含公钥摘要，实际公钥

可以被包含在SINIT-AC内或与SINIT-AC包含在一起。如上结合图2所述，一旦验证SINIT-AC的拷贝，ILP于是就可以在安全存储器208内实际执行SINIT-AC的拷贝。

[0052] SINIT-AC在安全存储器208内的拷贝开始执行之后，它随后(在时间周期640期间)确认并且登记SVMM的存储驻留拷贝。SVMM的拷贝登记在安全令牌276的PCR 278内之后，SVMM的存储驻留拷贝本身开始执行。此时，在正在进行的时间周期650期间，SVMM操作被建立在ILP内。

[0053] ILP SVMM操作最初要做的事情之一是在系统总线230上发布单独的RLP JOIN MESSAGES。一个实施例就是处理器222的JOIN MESSAGE 644。这个消息可以包括存储器内的一位置，在该位置上RLP处理器222可以加入被登记的SVMM存储驻留拷贝的执行。可选择地，ILP SVMM操作可能已经把存储器位置登记在芯片组或存储器内的预定位置中，一旦接收到JOIN MESSAGE，RLP就从所述位置取回它的开始地址。在接收到处理器222的JOIN MESSAGE并且确定它的开始地址之后，在时间周期646期间，RLP处理器222跳转到这个位置，并且加入被登记的SVMM存储驻留拷贝的执行。

[0054] 在所有RLP已经加入被登记的SVMM存储驻留拷贝之后，安全操作在整个微型计算机系统200上被建立起来。

[0055] 现在参考图7，表示依据本发明一个实施方案的软件和其它过程块(process block)的流程图。为了清楚，图7只表示用于单个有代表性的RLP的过程块。在其它实施方案中，可以存在几个响应逻辑处理器。

[0056] 过程700从过程块710开始，这时，逻辑处理器复制SINIT-AC和SVMM模块的拷贝，该SINIT-AC和SVMM模块的拷贝能由随后的SENTER指令所访问。在这个实施例中，在过程块712中，ILP把SINIT-AC和SVMM代码从大容量存储器加载到物理存储器内。在替代的实施方案中，任何逻辑处理器可以这样做，而不只是ILP。如在过程块714中所注，通过执行SENTER指令，处理器成为ILP。在过程块716中，ILP SENTER指令在过程块716中发布SENTER BUS MESSAGE。然后，ILP在过程块718中把它本身的SENTER ACK消息发送给芯片组。如判断过程块720所示，ILP于是进入等待状态，等待芯片组设定其ALL-JOINED标志。

[0057] 在每个RLP在过程块770中接收到SENTER BUS MESSAGE之后，它以当前指令的结束来暂停执行，然后在过程块772中发布它自己的SENTER ACK。如判断过程块774所示，每个RLP于是进入等待状态，等待从ILP到来的SENTER CONTINUE MESSAGE。

[0058] 当接收到SENTER ACK信息时，芯片组设定JOINS寄存器内的相应位。当JOINS寄存器的内容等于EXISTS寄存器的内容时，芯片组设定ALL-JOINED标志，给ILP发送信号以从判断过程块720继续进行。

[0059] 一旦在YES路径上离开判断过程块720，ILP于是就在过程块722中发布SENTER CONTINUE MESSAGE。这给每个RLP发送信号以从判断过程块774继续进行。如判断过程块776所示，然后每个RLP进入第二个等待状态，等待SENTER JOIN MESSAGE。

[0060] 同时，ILP在过程块724中把芯片组公钥和SINIT-AC存储驻留拷贝移入它自己的安全存储器用于安全执行。ILP在过程块726中使用所述密钥来验证SINIT-AC的安全存储驻留拷贝，然后执行它。SINIT-AC的执行可以进行对系统配置和SVMM拷贝的测试，然后登记SVMM身份，最后在过程块728中开始执行SVMM。作为在过程块728中执行的动作的一部分，ILP SINIT代码可以配置设备访问页面表248以及存储器和芯片组的设备访问逻辑247，以保护

SVMM 282存储驻留拷贝所使用的那些存储页面不受非处理器设备的干扰，如过程块754中所示。

[0061] ILP在SVMM控制下开始执行之后，在过程块730中，ILP给每个RLP发送单独的SENTER JOIN MESSAGE。发送SENTER JOIN MESSAGE之后，ILP随后在过程块732中开始SVMM操作。

[0062] SENTER JOIN MESSAGE的接收使每个RLP沿着YES路径离开由判断过程块776表示的等待状态，且在过程块780中开始SVMM操作。SENTER JOIN MESSAGE可以包含SVMM入口点，RLP在加入SVMM操作时分支进入该入口点。可选择地，ILP SVMM代码可以把适当的RLP入口点登记在系统位置内(例如在芯片组内)，RLP一旦接收到SENTER JOIN MESSAGE就重新取回它。

[0063] 虽然公开的不同实施方案包括两个或多个处理器(逻辑或物理处理器)，但应该理解，以更详细的方式描述这样的多个处理器和/或多个线程系统以解释与对带有多个逻辑或物理处理器的系统进行安全防护相关联的增加的复杂性。在复杂程度较底的系统中可能有利的实施方案可以只使用一个处理器。在某些情况下，一个物理处理器可以是多个线程，从而可以包括多个逻辑处理器(因此具有所述的ILP和RLP)。然而，在其它情况下，可以使用单个处理器、单线程系统，并仍然利用所公开的安全处理技术。在上述情况下，可以没有RLP；然而，所述安全处理技术仍然起作用来减少以未授权方式能够窃取或操纵数据的可能性。

[0064] 在前述说明书中，已经参考本发明的示例性实施方案对其进行了描述。然而，显然可以对本发明进行各种修改和变化而不脱离附属权利要求书所提出的本发明更宽的本质和范围。因此把说明书和附图看作例证性的而不是限制性的。

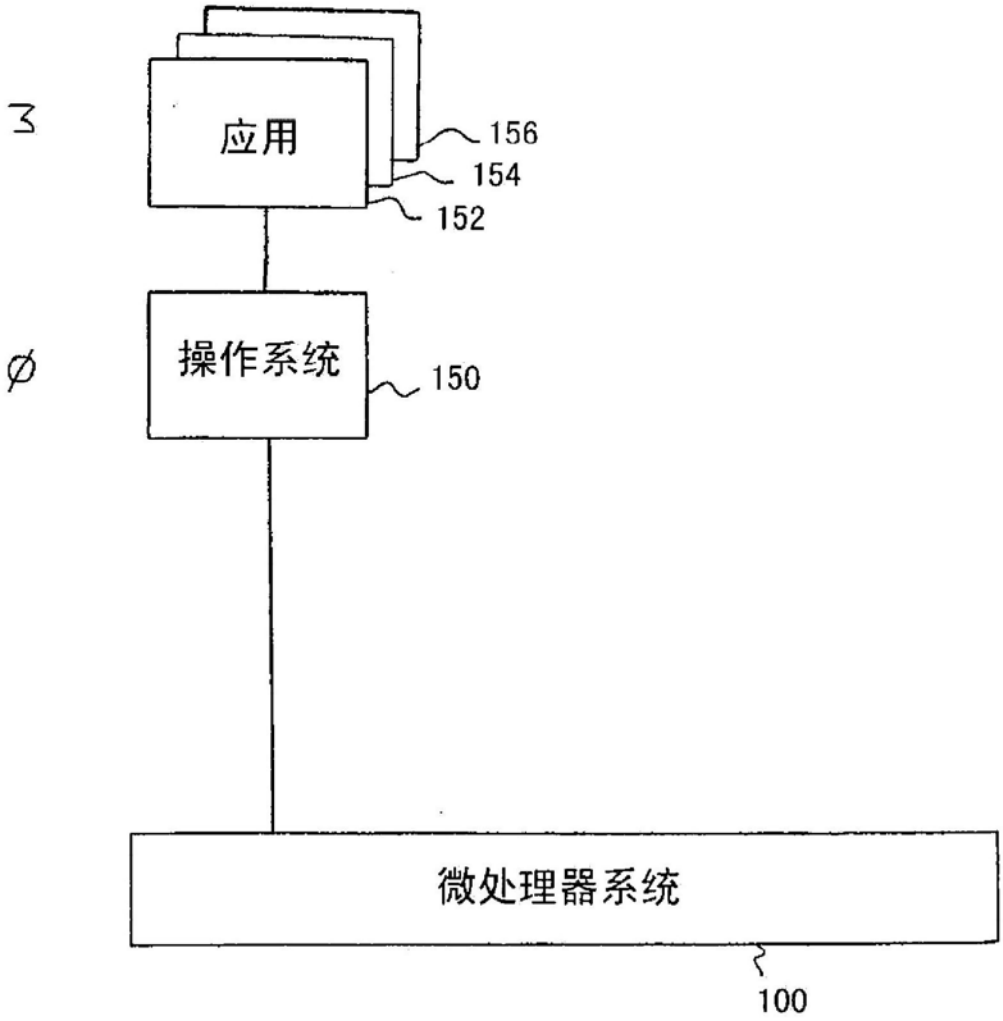


图1

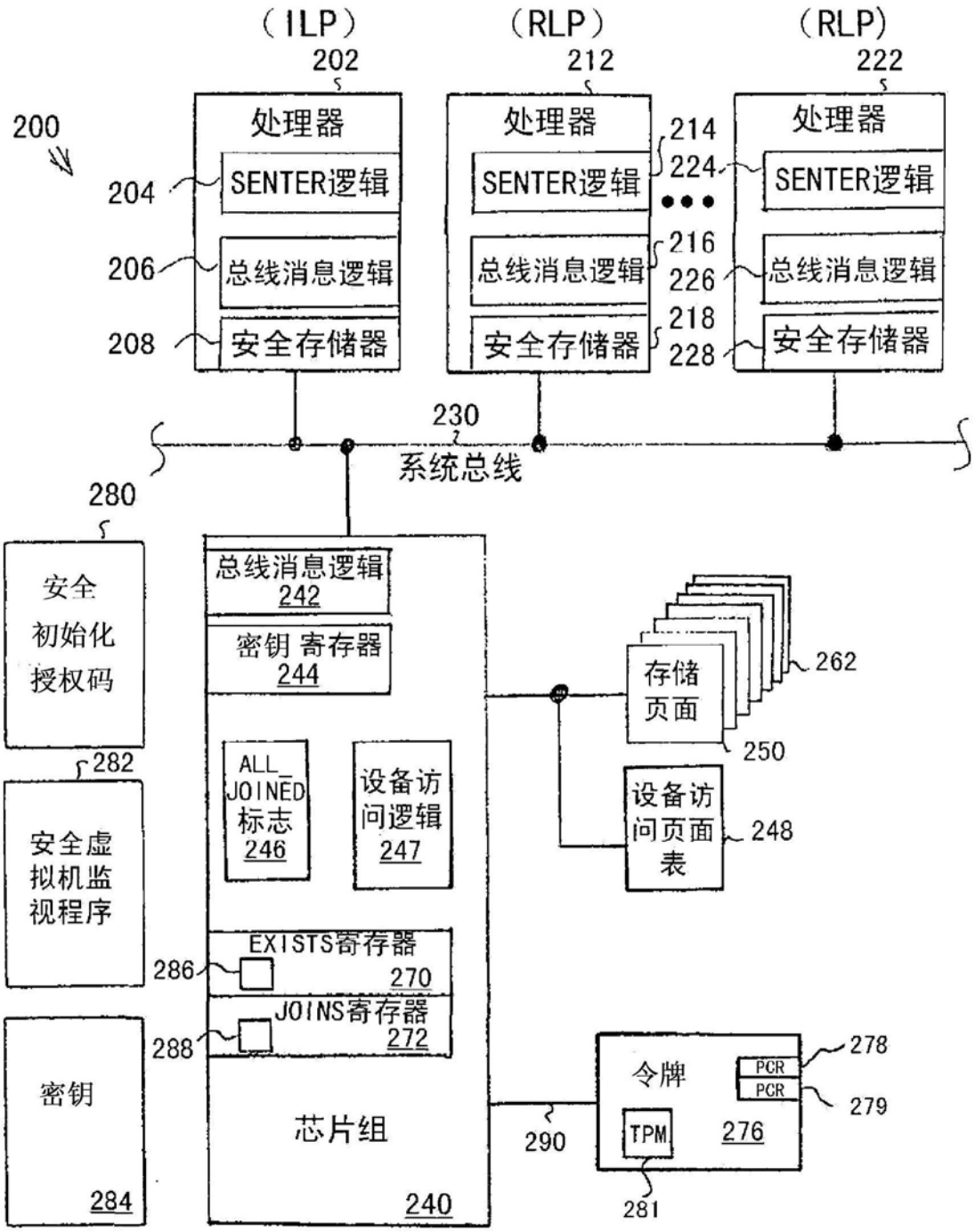

(ILP)
202
(RLP)
212
(RLP)
222
200
处理器
处理器
处理器
204
SENTER逻辑
SENTER逻辑
214
224
SENTER逻辑
206
总线消息逻辑
总线消息逻辑
216
226
总线消息逻辑
208
安全存储器
安全存储器
218
228
安全存储器
230
系统总线
280
安全初始化授权码
282
安全虚拟机监视程序
密钥
284
总线消息逻辑 242
密钥 寄存器 244
ALL_JOINED 标志 246
设备访问逻辑 247
EXISTS寄存器 270
286
JOINS寄存器 272
288
芯片组 240
存储页面
262
250
设备访问页面表
248
令牌
PCR
PCR
278
279
TPM
276
281
290

图2

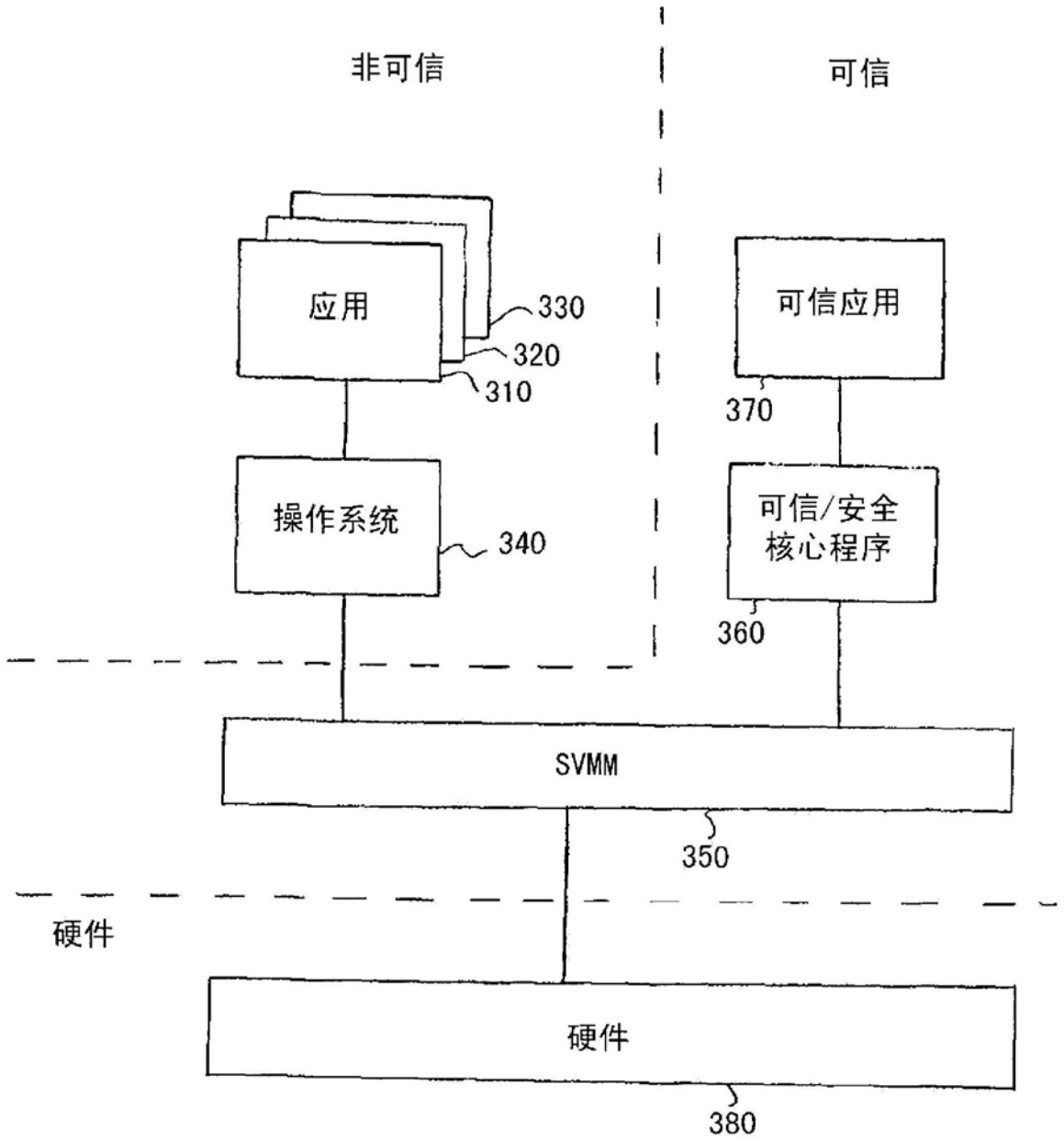

非可信
可信
应用
330
320
310
可信应用
370
操作系统
340
可信/安全
核心程序
360
SVMM
350
硬件
硬件
380

图3

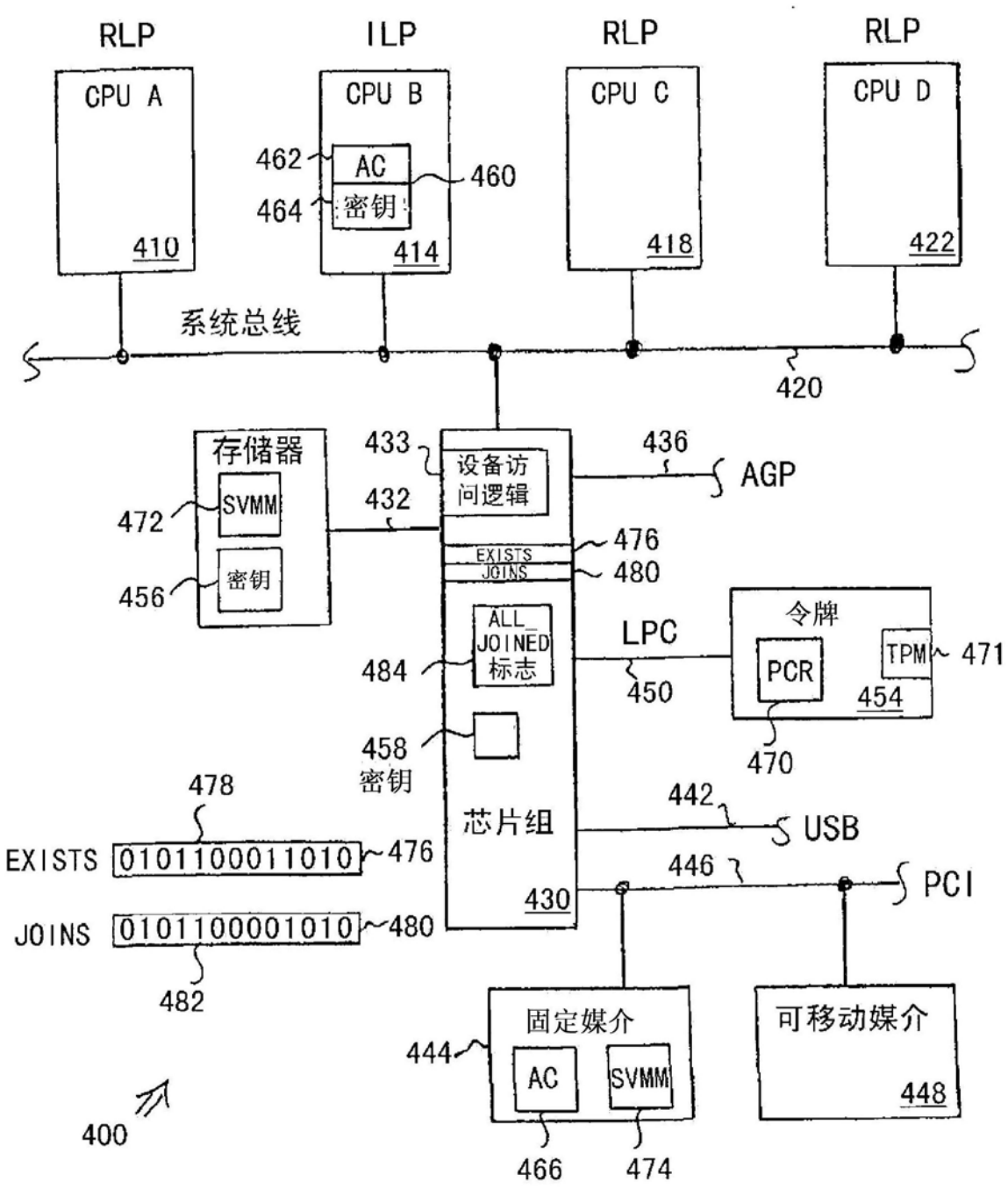

RLP
ILP
RLP
RLP
CPU A
410
CPU B
462
AC
460
464
密钥
414
CPU C
418
CPU D
422
系统总线
420
存储器
472
SVMM
456
密钥
433
设备访问逻辑
432
436
AGP
476
EXISTS
480
JOINS
ALL_JOINED标志
484
LPC
450
令牌
PCR
TPM
471
454
470
458
密钥
478
442
USB
芯片组
430
446
PCI
EXISTS
0101100011010
476
JOINS
0101100001010
480
482
固定媒介
444
AC
SVMM
466
474
可移动媒介
448
400

图4A

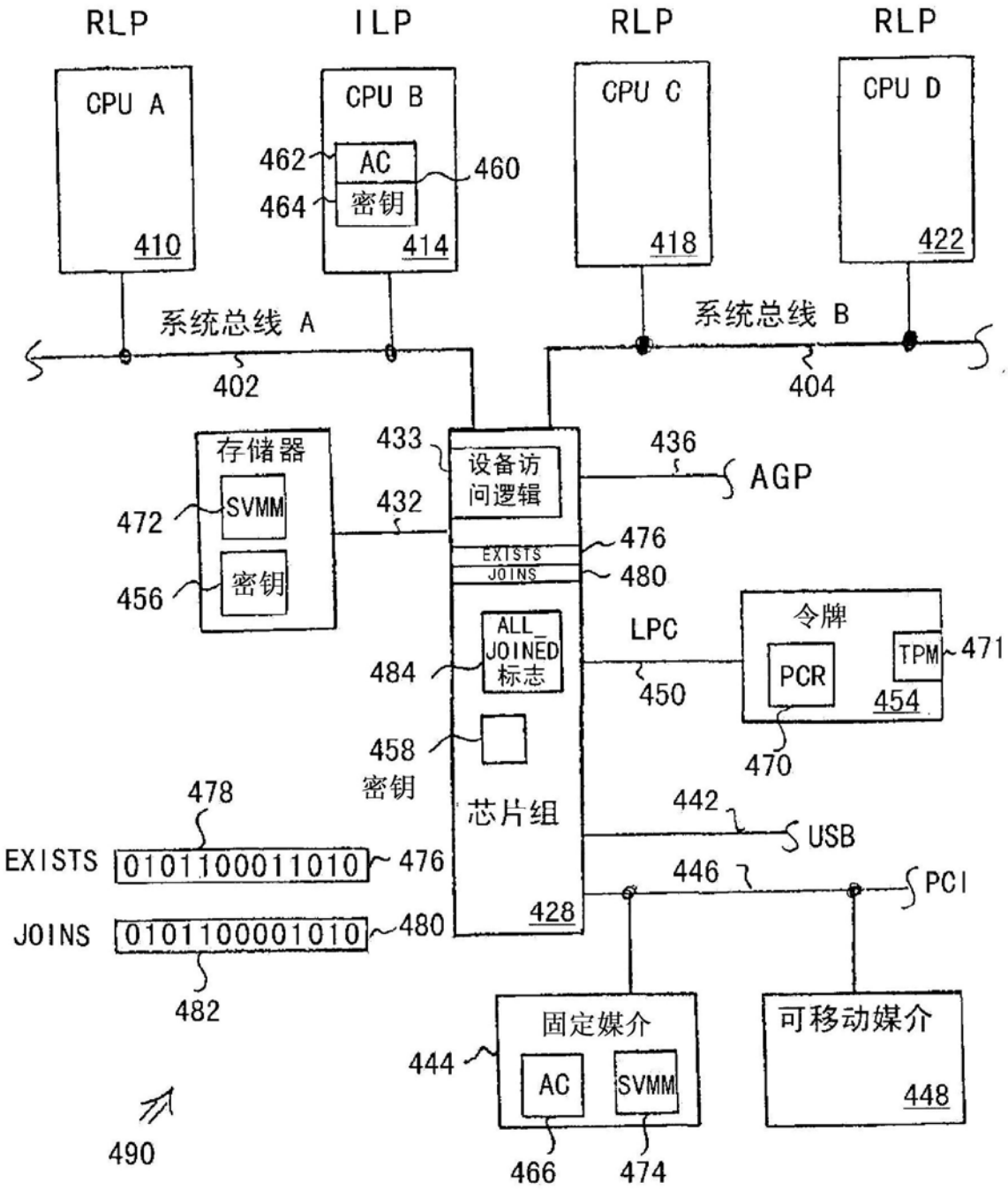

RLP
ILP
RLP
RLP
CPU A
410
CPU B
462
AC
460
464
密钥
414
CPU C
418
CPU D
422
系统总线 A
402
系统总线 B
404
存储器
472
SVMM
456
密钥
433
设备访问逻辑
432
436
AGP
476
EXISTS
480
JOINS
ALL_JOINED 标志
484
LPC
450
令牌
TPM
471
PCR
454
470
458
密钥
芯片组
428
478
EXISTS
0101100011010
476
JOINS
0101100001010
480
482
442
USB
446
PCI
固定媒介
444
AC
SVMM
466
474
可移动媒介
448
490

图4B

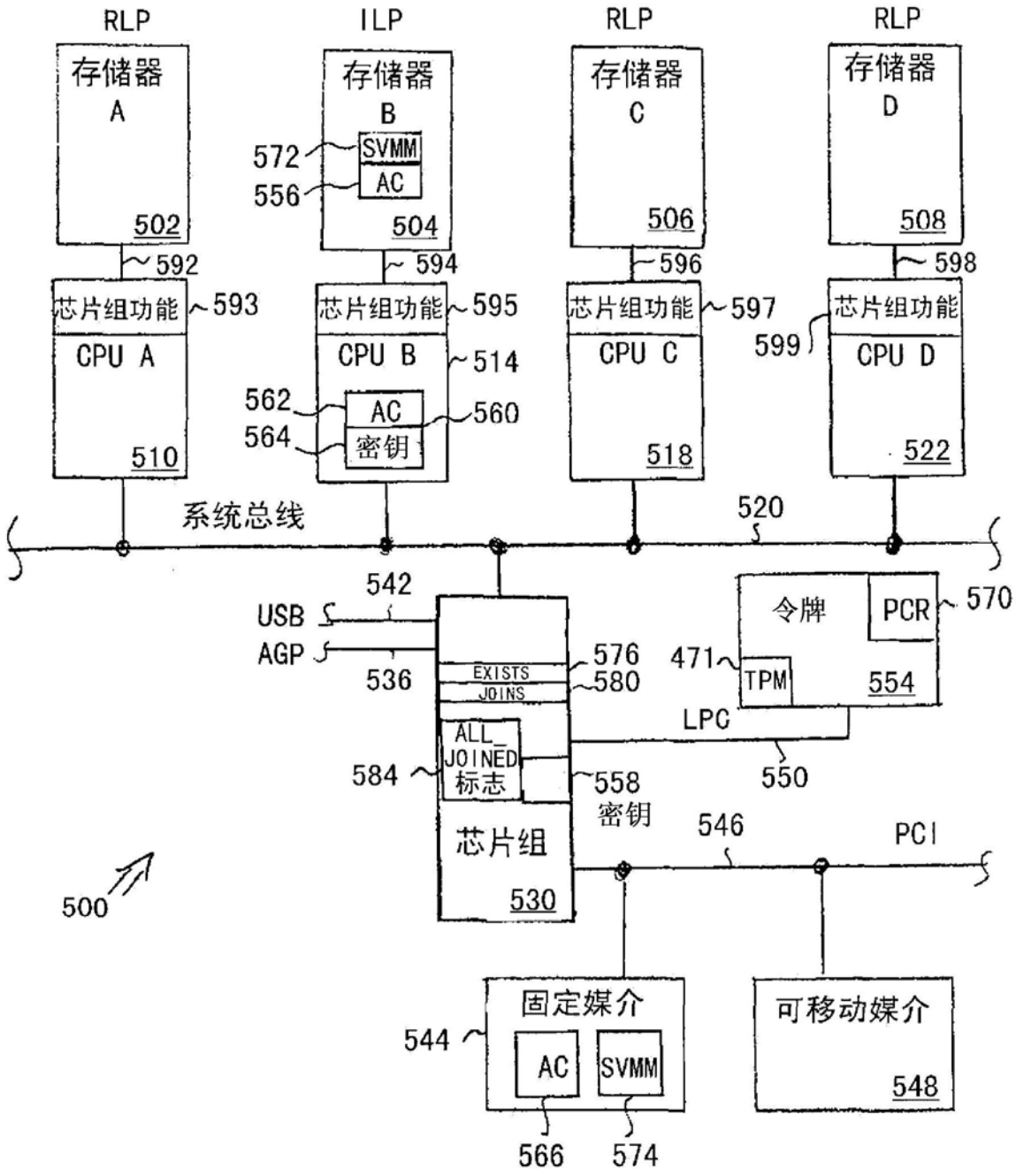

RLP
存储器
A
502
592
芯片组功能
593
CPU A
510
ILP
存储器
B
572
SVMM
556
AC
504
594
芯片组功能
595
CPU B
514
562
AC
560
564
密钥
RLP
存储器
C
506
596
芯片组功能
597
CPU C
518
RLP
存储器
D
508
598
599
芯片组功能
CPU D
522
系统总线
520
542
USB
AGP
536
EXISTS
576
JOINS
580
ALL_
JOINED
标志
584
芯片组
530
500
令牌
PCR
570
471
TPM
554
LPC
550
558
密钥
546
PCI
固定媒介
544
AC
SVMM
566
574
可移动媒介
548

图5

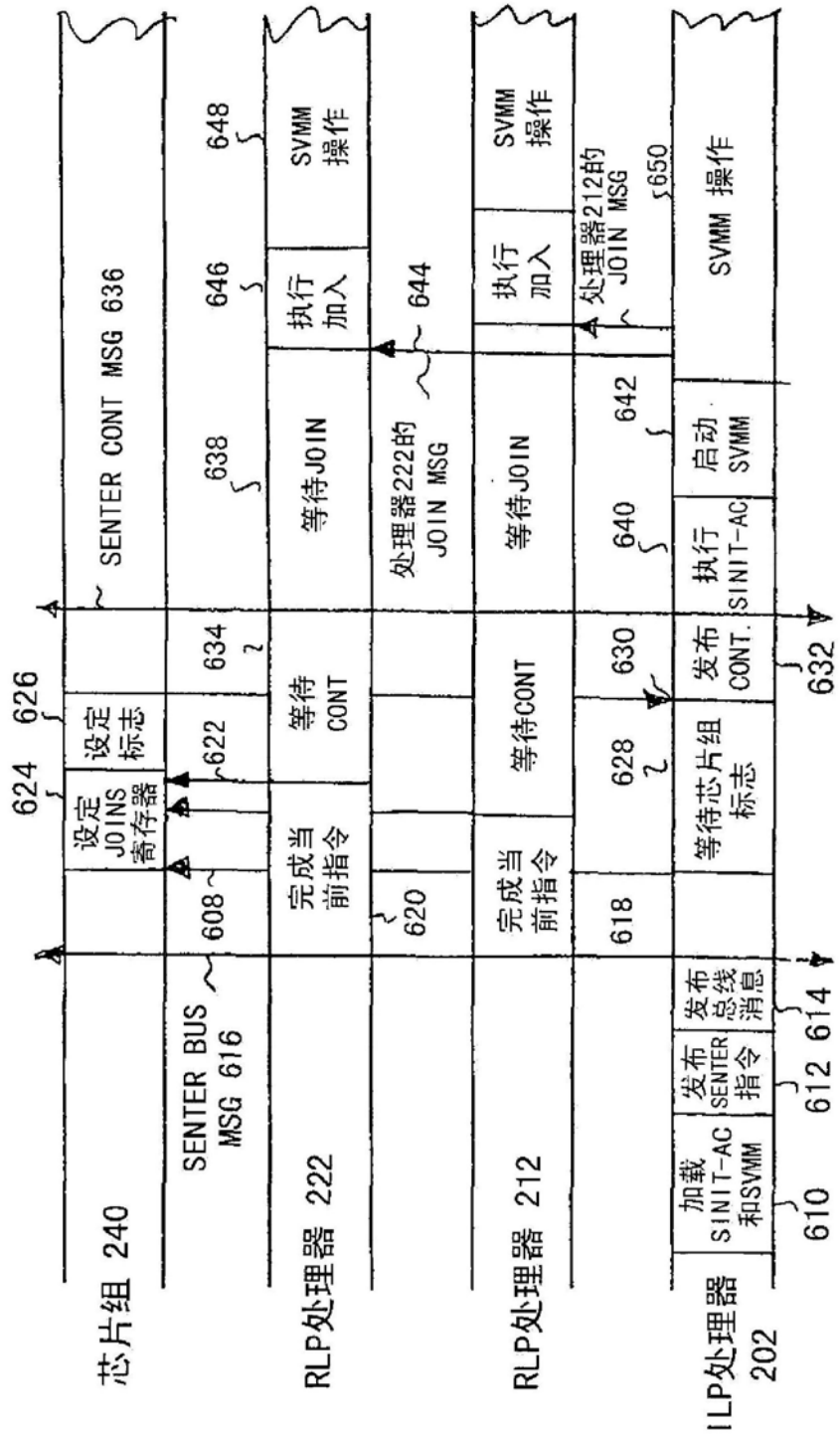

芯片组 240
RLP处理器 222
RLP处理器 212
ILP处理器 202
SENTER BUS MSG 616
SENTER CONT MSG 636
设定 JOINS 寄存器
设定 标志
608
622
634
638
646
648
624
626
完成当前指令
620
等待 CONT
等待JOIN
执行 加入
SVMM 操作
处理器222的 JOIN MSG
644
完成当前指令
618
等待CONT
628
630
等待JOIN
执行 加入
SVMM 操作
处理器212的 JOIN MSG
640
642
650
等待芯片组 标志
发布 CONT.
执行 SINIT-AC
启动 SVMM
SVMM 操作
加载 SINIT-AC 和SVMM
发布 SENTER 指令
发布总线消息
610
612
614
632

图6

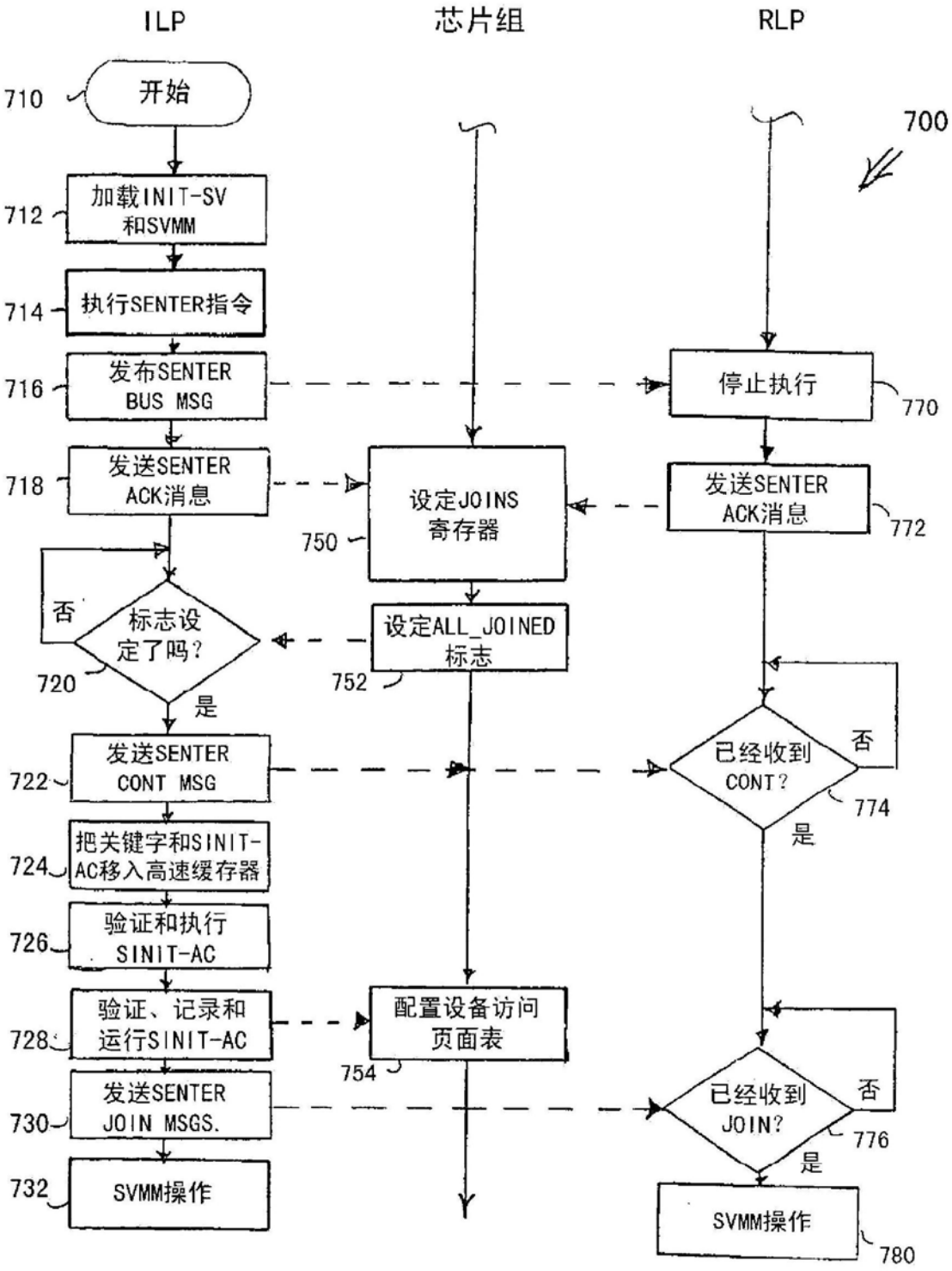

ILP
芯片组
RLP
700
710
开始
712
加载INIT-SV和SVMM
714
执行SENTER指令
716
发布SENTER BUS MSG
718
发送SENTER ACK消息
720
标志设定了吗?
否
是
722
发送SENTER CONT MSG
724
把关键字和SINIT-AC移入高速缓存器
726
验证和执行SINIT-AC
728
验证、记录和运行SINIT-AC
730
发送SENTER JOIN MSGS.
732
SVMM操作
750
设定JOINS寄存器
752
设定ALL_JOINED标志
754
配置设备访问页面表
770
停止执行
772
发送SENTER ACK消息
774
已经收到CONT?
否
是
776
已经收到JOIN?
否
是
780
SVMM操作

图7