(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 112449698 A
(43)申请公布日 2021.03.05

(21)申请号 201980048369.0

(22)申请日 2019.06.06

(30)优先权数据
1812314.1 2018.07.27 GB

(85)PCT国际申请进入国家阶段日
2021.01.19

(86)PCT国际申请的申请数据
PCT/GB2019/051566 2019.06.06

(87)PCT国际申请的公布数据
WO2020/021224 EN 2020.01.30

(71)申请人 ARM有限公司
地址 英国剑桥

(72)发明人 托马斯·克里斯多夫·格鲁卡特 弗朗索瓦·克里斯托弗·雅克·波特曼

(74)专利代理机构 北京东方亿思知识产权代理有限责任公司 11258
代理人 郭妍

(51)Int.Cl.
G06F 16/903(2006.01)

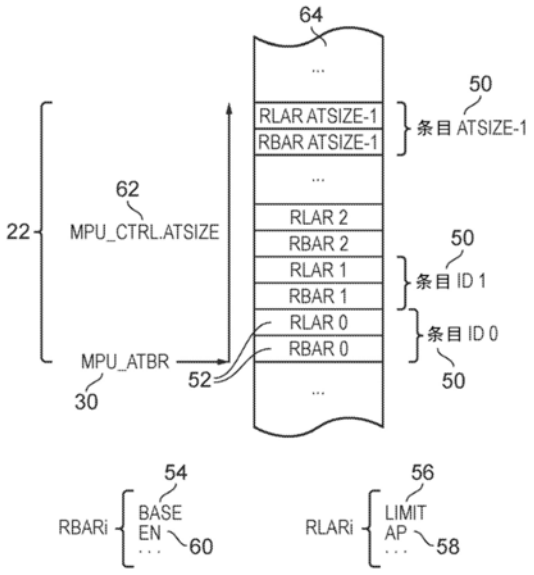
权利要求书3页 说明书36页 附图12页

(54)发明名称

针对存储在存储器系统中的控制表的二分搜寻过程

(57)摘要

控制表22定义了用于控制处理组件20执行操作的信息。表22包括下述条目:每个条目对应于由第一界限地址以及第二界限地址和大小中的一者定义的可变大小区域。提供了一种用于查找表的二分搜寻过程,该过程包括多个搜寻窗口窄化步骤,每个步骤搜寻窗口窄化基于查询地址与当前搜寻窗口的所选择的候选条目的第一界限地址进行的比较,来将候选条目的当前搜寻窗口窄化为包括更少条目的更窄搜索窗口。该比较与所选择的候选条目的第二界限地址或大小无关。在将搜寻窗口窄化为单个条目之后,将查询地址与该单个条目的第二界限地址或大小进行比较。



CN 112449698 A

1.一种装置,包括:

处理组件,用以基于存储在存储器系统中的控制表中指定的信息来执行操作,所述控制表包括多个条目,每个条目指定了第一界限地址以及第二界限地址和大小中的一者,用于标识地址空间内可变大小的对应地址区域的界限,其中所述多个条目依照第一界限地址的顺序被布置在所述控制表中;以及

表查找电路,用以执行二分搜寻过程以标识所述控制表是否包括对应于查询地址的条目,所述二分搜寻过程包括多个搜寻窗口窄化步骤,用于将所述控制表的候选条目的当前搜寻窗口窄化为包括更少候选条目的更窄搜寻窗口;其中:

在给定搜寻窗口窄化步骤中,所述表查找电路被配置为:独立于所述当前搜寻窗口的所选择的候选条目的第二界限地址和大小中的所述一者,基于所述查询地址与所选择的候选条目的第一界限地址进行的比较来窄化所述当前搜寻窗口;并且

在将所述当前搜寻窗口窄化为所述控制表中的单个候选条目的最终搜寻窗口窄化步骤之后,所述表查找电路被配置为:当所述查询地址在由所述单个候选条目的第一界限地址以及第二界限地址和大小中的所述一者定义的对应地址区域内时,所述单个候选条目是对应于所述查询地址的条目。

2.根据权利要求1所述的装置,其中,在所述二分搜寻过程中,在所述最终搜寻窗口窄化步骤之后标识的所述单个候选条目是所述控制表中唯一的下述条目:针对该条目,所述表查找电路触发存储器访问请求以便从所述存储器系统获得第二界限地址和大小中的所述一者,来作为针对所述查询地址执行所述二分搜寻过程的结果。

3.根据权利要求1和2中任一项所述的装置,其中,当所述查询地址在由所述单个候选条目的第一界限地址以及第二界限地址和大小中的所述一者定义的对应地址区域之外时,所述表查找电路被配置为确定所述查询地址在所述控制表中未命中。

4.根据权利要求3所述的装置,其中,当所述查询地址在所述控制表中未命中时,所述表查找电路被配置为执行以下步骤中的至少一项:

对异常情况进行信号告知;并且

返回预设信息来作为用于控制所述操作的信息。

5.根据任一前述权利要求中所述的装置,其中,在所述给定搜寻窗口窄化步骤中,在所述比较指示所述查询地址在所选择的候选条目的对应地址区域之外时,所述表查找电路被配置为:

用被包括在所述更窄搜寻窗口中的所选择的候选条目来窄化所述当前搜寻窗口;或者

用被排除在所述更窄搜寻窗口外的所选择的候选条目来窄化所述当前搜寻窗口。

6.根据任一前述权利要求中所述的装置,包括:用以存储至少一个条目的条目存储装置,

其中所述条目存储装置包括以下各项中的一项:

条目缓存,用以缓存先前从所述存储器系统获得的所述控制表中的至少一个条目;以及

静态寄存器存储装置,用以存储能通过静态条目参数配置操作配置的至少一个另外的条目,每个另外的条目指定了所述地址空间内可变大小的下述对应地址区域的控制信息:该对应地址区域的界限通过第一界限地址以及第二界限地址和大小中的一者进行标识。

7.根据权利要求6所述的装置,其中,所述表查找电路被配置为:检查所述查询地址是否对应于存储在所述条目存储装置中的所述至少一个条目,并且根据所述查询地址是否对应于存储在所述条目存储装置中的任何条目来确定是否执行所述二分搜寻过程。

8.根据权利要求6和7中任一项所述的装置,其中,所述表查找电路被配置为:基于从存储在所述条目存储装置中的所述至少一个条目推导出的信息,来限制用于所述二分搜寻过程的初始搜寻窗口窄化步骤的当前搜寻窗口。

9.根据权利要求6至8中任一项所述的装置,其中,在所述二分搜寻过程的给定二分搜寻步骤中,当所述查询地址与所选择的候选条目的第一界限地址的比较结果能从存储在所述条目存储装置中的所述至少一个条目中的信息推导出时,所述表查找电路被配置为抑制用于从所述存储器系统获得所选择的候选条目的第一界限地址的存储器访问请求。

10.根据任一前述权利要求中所述的装置,其中,当所述控制表包括N个条目时,其中N并非是精确的2的幂的数,所述表查找电路被配置为:执行所述二分搜寻过程,其中假设初始搜寻窗口窄化步骤的当前搜寻窗口所包括的候选条目数对应于大于N的最小的2的幂。

11.根据权利要求10所述的装置,其中,在其中所选择的候选条目对应于所述控制表的所述N个条目的范围之外的位置的搜寻窗口窄化步骤中,所述表查找电路被配置为:抑制用于从所述存储器系统获得所选择的候选条目的第一界限地址的存储器访问请求,并且将下述隐含结果确定为所述比较的结果:所述隐含结果使得更窄搜寻窗口包括有所在位置位于所述控制表的所述N个条目的范围之内的至少一个候选条目。

12.根据任一前述权利要求所述的装置,其中,对于每个条目,所述查表电路被配置为允许以下各项中的至少一项:

可变大小为不是2的幂的字节数;以及

在对应地址区域的起始地址和结束地址能够从第一界限地址以及第二界限地址和大小中的所述一者标识出的情况下,使对应地址区域的起始地址和结束地址不与X字节地址边界对准,其中X是对应地址区域的大小。

13.根据任一前述权利要求所述的装置,其中,所述第一界限地址包括对应地址区域的起始地址。

14.根据权利要求1至12中任一项所述的装置,其中,所述第一界限地址包括对应地址区域的结束地址。

15.根据任一前述权利要求所述的装置,其中,对于所述控制表中的给定条目,第一界限地址被存储在所述存储器系统内与用来存储所述给定条目的第二界限地址和大小中的所述一者的数据字不同的数据字中。

16.根据任一前述权利要求所述的装置,其中:

所述控制表包括存储器保护表,所述存储器保护表的每个条目指定了地址空间中的对应地址区域的访问权限;并且

基于所述控制表中指定的信息执行的所述操作包括:检查指定所述查询地址的存储器访问请求是否满足所述访问权限。

17.一种方法,包括:

执行二分搜寻过程以标识存储在存储器系统中的控制表是否包括对应于查询地址的条目,所述控制表包括多个条目,每个条目指定了第一界限地址以及第二界限地址和大小

中的一者，用于标识地址空间内可变大小的对应地址区域的界限，其中所述多个条目依照第一界限地址的顺序布置在所述控制表中；并且

当所述控制表包括有对应于所述查询地址的条目时，基于在对应于所述查询地址的条目中指定的信息来执行操作；其中：

所述二分搜寻过程包括多个搜寻窗口窄化步骤，用于将所述控制表的候选条目的当前搜寻窗口窄化到包括更少候选条目的更窄搜寻窗口；

在给定搜寻窗口窄化步骤中，独立于当前搜寻窗口的所选择的候选条目的第二界限地址和大小中的所述一者，基于所述查询地址与所选择的候选条目的第一界限地址进行的比较来窄化所述当前搜寻窗口；并且

在将所述当前搜寻窗口窄化为所述控制表中的单个候选条目的最终搜寻窗口窄化步骤之后，当所述查询地址在由所述单个候选条目的第一界限地址以及第二界限地址和大小中的所述一者定义的对应地址区域内时，所述单个候选条目被确定为对应于所述查询地址的条目。

18.一种计算机程序，用于控制主机数据处理装置以提供供指令执行的指令执行环境；所述计算机程序包括：

处理程序逻辑，用以基于控制表中指定的信息来执行操作，所述控制表包括多个条目，每个条目指定了第一界限地址以及第二界限地址和大小中的一者，用于标识地址空间内可变大小的对应地址区域的界限，其中所述多个条目依照第一界限地址的顺序布置在所述控制表中；以及

表查找程序逻辑，用以执行二分搜寻过程以标识所述控制表是否包括对应于所述查询地址的条目，所述二分搜寻过程包括多个搜寻窗口窄化步骤，用于将所述控制表的候选条目的当前搜寻窗口窄化为包括更少候选条目的更窄搜寻窗口；其中：

在给定搜寻窗口窄化步骤中，所述表查找程序逻辑被配置为：独立于所述当前搜寻窗口的所选择的候选条目的第二界限地址和大小中的所述一者，基于所述查询地址与所选择的候选条目的第一界限地址进行的比较来窄化所述当前搜寻窗口；并且

在将所述当前搜寻窗口窄化为所述控制表中的单个候选条目的最终搜寻窗口窄化步骤之后，所述表查找程序逻辑被配置为：当所述查询地址在由所述单个候选条目的第一界限地址以及第二界限地址和大小中的所述一者定义的对应地址区域内时，所述单个候选条目对应于所述查询地址的条目。

19.一种用于存储根据权利要求18所述的计算机程序的存储介质。

针对存储在存储器系统中的控制表的二分搜寻过程

技术领域

[0001] 本公开涉及数据处理领域。

背景技术

[0002] 数据处理装置可以具有下述处理组件:用于基于存储在存储器系统中的控制表中指定的信息来执行操作的处理组件。例如,控制表可定义控制是否允许某些软件进程访问地址空间的某些区域的访问权限。

发明内容

[0003] 至少一些示例提供了一种装置,包括:

[0004] 处理组件,用以基于存储在存储器系统中的控制表中指定的信息来执行操作,控制表包括多个条目,每个条目指定了第一界限地址以及第二界限地址和大小中的一者,用于标识地址空间内可变大小的对应地址区域的界限,其中多个条目依照第一界限地址的顺序被布置在控制表中;以及

[0005] 表查找电路,用以执行二分搜寻过程以标识控制表是否包括对应于查询地址的条目,二分搜寻过程包括多个搜寻窗口窄化步骤,用于将控制表的候选条目的当前搜寻窗口窄化为包括更少候选条目的更窄搜寻窗口;其中:

[0006] 在给定搜寻窗口窄化步骤中,表查找电路被配置为:独立于当前搜寻窗口的所选择的候选条目的第二界限地址和大小中的所述一者,基于查询地址与所选择的候选条目的第一界限地址进行的比较来窄化当前搜寻窗口;并且

[0007] 在将当前搜寻窗口窄化为控制表中的单个候选条目的最终搜寻窗口窄化步骤之后,表查找电路被配置为:当查询地址在由单个候选条目的第一界限地址以及第二界限地址和大小中的所述一者定义的对应地址区域内时,该单个候选条目是对应于查询地址的条目。

[0008] 至少一些示例提供了一种方法,包括:

[0009] 执行二分搜寻过程以标识存储在存储器系统中的控制表是否包括对应于查询地址的条目,控制表包括多个条目,每个条目指定了第一界限地址以及第二界限地址和大小中的一者,用于标识地址空间内可变大小的对应地址区域的界限,其中多个条目依照第一界限地址的顺序布置在控制表中;并且

[0010] 当控制表包括有对应于查询地址的条目时,基于在对应于查询地址的条目中指定的信息来执行操作;其中:

[0011] 二分搜寻过程包括多个搜寻窗口窄化步骤,用于将控制表的候选条目的当前搜寻窗口窄化到包括更少候选条目的更窄搜寻窗口;

[0012] 在给定搜寻窗口窄化步骤中,独立于当前搜寻窗口的所选择的候选条目的第二界限地址和大小中的所述一者,基于查询地址与所选择的候选条目的第一界限地址进行的比较来窄化当前搜寻窗口;并且

[0013] 在将当前搜寻窗口窄化为控制表中的单个候选条目的最终搜寻窗口窄化步骤之后,当查询地址在由单个候选条目的第一界限地址以及第二界限地址和大小中的所述一者定义的对应地址区域内时,单个候选条目被确定为对应于所述查询地址的条目。

[0014] 至少一些示例提供了一种计算机程序,用于控制主机数据处理装置以提供供指令执行的指令执行环境;该计算机程序包括:

[0015] 处理程序逻辑,用以基于控制表中指定的信息来执行操作,控制表包括多个条目,每个条目指定了第一界限地址以及第二界限地址和大小中的一者,用于标识地址空间内可变大小的对应地址区域的界限,其中多个条目依照第一界限地址的顺序布置在所述控制表中;以及

[0016] 表查找程序逻辑,用以执行二分搜寻过程以标识控制表是否包括适用于查询地址的条目,二分搜寻过程包括多个搜寻窗口窄化步骤,用于将控制表的候选条目的当前搜寻窗口窄化为包括更少候选条目的更窄搜寻窗口;

[0017] 其中:

[0018] 在给定搜寻窗口窄化步骤中,表查找程序逻辑被配置为:独立于当前搜寻窗口的所选择的候选条目的第二界限地址和大小中的所述一者,基于查询地址与所选择的候选条目的第一界限地址进行的比较来窄化当前搜寻窗口;并且

[0019] 在将当前搜寻窗口窄化为控制表中的单个候选条目的最终搜寻窗口窄化步骤之后,表查找程序逻辑被配置为:当查询地址在由单个候选条目的第一界限地址以及第二界限地址和大小中的所述一者定义的对应地址区域内时,该单个候选条目是对应于查询地址的条目。

附图说明

[0020] 本技术的其他方面、特征和优点将从对将结合附图来解读的示例的以下描述中显而易见,在附图中:

[0021] 图1示意性地图示出具有MPU的数据处理装置的示例;

[0022] 图2示出了存储在存储器系统中的存储器保护表的示例;

[0023] 图3示出了用于从存储器保护表缓存存储器保护条目的MPU缓存存储装置的示例;

[0024] 图4示出了MPU的寄存存储装置的示例;

[0025] 图5示出了基于存储器的MPU条目或静态MPU条目可如何涵盖地址空间的不同地址区域;

[0026] 图6是示出了使用MPU来控制对存储器系统的访问的方法的流程图;

[0027] 图7是示出了基于二分搜寻过程查找控制表的方法的流程图;

[0028] 图8示出了针对包括高达16个条目的表的二分搜寻过程的具体示例;

[0029] 图9示出了二分搜寻过程的样例;

[0030] 图10示出了惰态(lazy state)保存的示例;

[0031] 图11A到图12B示出了图示出控制惰态保存的示例的流程图;并且

[0032] 图13示出了可以使用的模拟器示例。

具体实施方式

[0033] 对存储器的访问的控制

[0034] 一种用于控制对存储器系统的访问的方法可以是提供存储器管理单元(MMU),该存储器管理单元检查存储器访问请求是否满足存储在存储器系统中的页表中指定的访问权限。地址空间通常被分成称为页面的区域,每个页面具有与二的一定幂的字节数对应的大小(例如4千字节),其中每个页面的起始地址和结束地址与N字节地址边界对准,其中N是相应地址区域的大小。每个页面条目指定了地址空间中相应页面的访问权限,并且通常还会指定地址转换映射,用于将第一地址空间(例如虚拟地址空间)中的目标地址映射到第二地址空间(例如物理地址空间)中的对应地址页面。可以将页表实现为包括许多页表等级的层级表结构,其中输入地址的第一位子集对第一级页表中的相应条目加索引,并且具有索引的条目指定了第二级页表的基础地址,可以根据该基础地址并基于输入地址的第二位子集对条目加索引,针对每个后续等级的表以这样的推,直到最后到达最终等级的页表为止,在该最终等级的页表处,具有索引的条目是基于指定的一些地址位选择的。具有索引的条目包含将该页面作为目标的存储器访问所需的实际地址转换属性和访问权限。MMU可以具有缓存(称为转换后备缓冲器(TLB)),该缓存可以缓存页表的最近更多地从存储器系统访问的某些条目,使得在处理电路发出要求相同的页面条目的多个存储器访问请求时可以更快速地再次访问这些条目。MMU可以用于被设计为处理相对高性能的工作负载的处理器,对此,设置针对对许多不同的地址区域的访问的控制是可接受的(例如这样的控制可以在4千字节的粒度进行),并且对此,可能需要从第一地址空间中的任何页面到第二地址空间中的任何页面的任意地址转换映射,以便处理由许多不同软件进程(它们使用相同的虚拟地址但需要被映射到由存储器系统所使用的不同物理地址)触发的访问。

[0035] 然而,对于针对更具能源效率的工作负载或被设计为用于实时应用的处理器而言,MMU可能并不适合,因为用来定义对各个页面的访问权限的大型页表结构和控制的粒度对于通常在针对更具能源效率的较低性能的应用的系统(可能只有两千字节的随机访问存储器(RAM))中所见的受限存储器环境而言可能太大。此外,为了促进将输入地址位的子集用作页表的索引,MMU通常将每个页面限制为具有与二的幂的字节数对应的大小。这意味着,如果需要仅针对具有二的幂的字节数以外的大小的某个地址区域设置某一访问权限组,则这会需要使用许多单独的页面条目(每一者与具有二的幂的字节数的大小的较小页面相对应)来表示,其中这些条目中的每一者准确地定义了相同的访问权限,使得作为一个整体的页面条目组与较大的非二的幂的大小的地址区域相对应。如此将地址区域人为分割成多个页面是限制页面与跟页面大小对应的地址边界对准(需要这样的限制来实现高效加索引过程,在该加索引过程中使用来自输入地址的全部位数来对用以涵盖大型地址空间的层级页面结构加索引)的结果。

[0036] 然而,对于实时代码,该方法会引起一些问题,因为这意味着,在旨在由给定的访问权限组所支配的代码段或数据内,存储器访问可能跨越与页表的不同页面条目对应的页面之间的页面边界,并且因此如果新页面的页面条目还未被缓存在MMU中,则这可能需要在执行代码段中途执行页表查找以从存储器取出所需的页面条目。此外,如果在到达页面边界时,发现下个页面的页面条目还未被定义在页表中,则这可能引发错误,对该错误可能处理缓慢,因为可能需要软件异常处理程序触发操作系统或其他监督进程来设置页面条目以

映射所需的页面，从而造成长的延迟。这样的延迟在时长上可能是高度不同的，并且在最差的情况下可能时间极长，并且如果代码段对跨越多于两个不同的页面的地址发出存储器访问，则在每个页面边界处，可能再次招致这样的延迟。这使得MMU非常难以能够满足实时应用中具有确定响应时间的需求。

[0037] 另一种用于控制对存储器的访问的方法可以是提供存储器保护单元(MPU)，该存储器保护单元检查由处理电路发出的存储器访问请求是否满足许多存储器保护条目中指定的访问权限。每个存储器保护边界可对应于与地址空间内可变大小的特定地址区域。给定存储器保护条目的对应地址区域可以以起始地址和结束地址(通过给定存储器保护条目的参数来标识)为界。与由MMU使用的页表不同，对于MPU而言，与存储器保护条目对应的区域可以具有可变的大小(可以被允许是非二的幂的字节数)。利用此方法，可以保证，可以使用单个存储器保护条目来定义任意(非2的幂)大小的存储器地址区域，以避免每次在地址访问跨越页面地址边界时发生的页面查找与页面错误的风险。这使得MPU更适于被设计为用于处理实时应用的系统。此外，MPU通常将存储器保护条目中的每一者存储在MPU内本地提供的寄存器内，这些寄存器可以被快速访问，从而再次协助实时处理并避免需要将具有层级结构的大型页表存储在存储器系统内，这对于具有有限RAM的系统而言可以是重要的。

[0038] 然而，在包括MPU的系统中，缺点在于：因为用于控制对存储器的访问的配置数据是使用通常直接连接到并联比较器列的寄存器来指定的，这些比较器用于将要被访问的目标地址与存储在寄存器中的每个存储器保护条目的区域标识参数进行比较，所以与MPU相关联的功率和面积随着由MPU所支持的存储器保护条目数量的增加而严重地成比例增加。实际上，这可能意味着，所支持的存储器保护条目的数量可能限于相对小的数量，例如8个或16个。典型MPU中有限数量的存储器保护条目可能对由在处理系统上执行的软件所实现的性能造成严重影响。

[0039] 例如，在由MPU所支持的保护条目的数量上，与通过硬件支持的相比，一部分软件可能需要将较大数量的不同地址区域定义为具有不同的访问权限。在此情况下，在任何给定的时间，由该软件进行访问所需的地址空间的至少一些部分当前并不具有针对这些部分所定义的对应存储器保护条目。这会意味着，对地址空间的当前未定义的区域中的地址的访问会触发异常，该异常会接着需要软件干预并重新配置MPU的寄存器，以便用现在访问所需要的区域所需的参数覆写先前针对不同区域所定义的参数。如果之后的指令接着访问已经被覆写的先前定义的区域，这可能接着导致MPU配置切换回来。MPU寄存器的内容的这样的不断切换(每次都需要异常来触发切换)极大地损害了性能。

[0040] 另一示例可以是，处理器需要执行互相不信任的多个软件部分，但这些软件部分需要在它们之间共享数据。为了实现数据的共享，给定的软件部分可能希望分割地址空间的各部分，使得只有地址空间中的有限部分可被其他进程访问，但为了针对对地址空间的访问提供这样的精细的控制程度，这可能需要定义额外的地址区域，并且可能没有足够的由硬件中的MPU支持的空闲存储器保护条目来允许这样的精细粒度的控制。因此，实际上，相同的数据可能最后被复制到地址空间的多个部分中，一个复本在由第一软件访问的部分中，并且另一复本在由第二软件访问的部分中，这造成了在执行复制上的时间浪费和对可用的存储器容量的低效使用，这在具有有限存储器资源的系统中是一个较特殊的问题。

[0041] 因此，典型的MMU和MPU目前不提供在具有实时需求和/或相对有限的存储器容量

的系统中基于每区域定义的访问权限来控制对存储器系统的访问的高效的解决方案。

[0042] 具有基于存储器的存储器保护表的MPU

[0043] 在下文讨论的技术中,提供了一种MPU,该MPU包括权限检查电路,该权限检查电路用以检查由处理电路发出的存储器访问请求是否满足存储在存储器系统中的存储器保护表中指定的访问权限。存储器保护表包括许多存储器保护条目,每个存储器保护条目指定了地址空间内可变大小的对应地址区域的访问权限。每个条目还指定了区域标识参数,这些区域标识参数用于标识对应地址区域的起始地址和结束地址。MPU允许可变大小为并非2的幂的字节数。提供了MPU存储器访问电路,以发起至少一个存储器访问请求以便从存储器系统访问存储器保护表。

[0044] 因此,通过在访问受MPU管辖的相同存储器系统中提供存储器保护表,这提供了对提供较大数量的存储器保护条目的灵活性,这对于典型MPU(其基于针对每个条目存储并行地与目标地址进行比较的区域定义数据的寄存器)是符合实际的。可能是违反直觉的是,将存储器保护表移动到存储器系统(对该存储器系统的访问通常比寄存器慢)中会提供改进的软件性能。然而,虽然从存储器系统访问单个存储器保护条目可能是较慢的,但可以通过避免上文讨论的用于处理由硬件中的MPU支持的条目数量不足以处理软件需求的情况的软件变通方案(workaround)(否则会由于遭遇到由对地址空间的未定义区域的访问所引起的频繁异常而导致较大性能成本)来改进整体性能。利用存储器中提供的存储器保护表,硬件成本随条目数量的增加的增加要缓慢得多。

[0045] 此外,通过允许每个区域具有对应于并非2的幂的字节数的任意大小,这意味着,每个区域可以涵盖大量的存储器,并且无需将旨在具有相同访问权限的区域细分成多个单独的条目,这样的细分会导致在地址跨越区域边界时造成存储器保护错误的风险。这使得MPU在服务实时应用(其需要以快速和/或确定响应时间来处理关键的操作)的时比MMU更佳。

[0046] 存储器保护表的条目可以按照每个条目的对应地址区域的起始地址的顺序布置在地址空间中。这可能与MMU使用的页表结构不同,对于该页表结构而言,页表的每个等级可以包括具有指向另一等级的页表的指针的条目,并且因此在整个页表结构内,可以将最终等级(其实际提供相应页面的访问权限)的页面条目以与每个对应地址页面的起始地址的顺序不同的顺序布置在地址空间内。通过将存储器保护表的存储器保护条目实现为按照起始地址的顺序布置的线性访问表,这可以使得针对给定的目标地址对存储器保护表对相关的存储器保护条目的搜寻更简单。

[0047] 每个存储器保护条目对应于地址空间的给定地址区域。在一些实施方式中,地址空间可以是物理地址空间。因此,可以基于由被处理电路发出的存储器访问请求指定的物理地址来查询存储器保护表,其中物理地址与存储器系统中被访问的对应位置直接对应。这可以适用于实时应用,因为通过避免地址转换,这可以使得对存储器的访问更快速,这是因为可以与对存储器保护表的访问并行地执行对存储器的访问。如果随后确定权限不被满足,则由对存储器的访问所返回的数据不会被转发到处理电路。

[0048] 在MPU存储器访问电路发起至少一个MPU存储器访问请求以便从存储器系统访问存储器保护表时,这可能需要访问存储器保护表的多个存储器保护条目,而不是仅访问与包括目标地址的地址区域对应的存储器保护条目。这是因为,在每个条目能够与非二的幂

的字节数的可变大小区域对应时，可能事先并不知道存储器系统中与给定地址区域相关联的条目所在的地址，并且因此实际可能需要对该表进行搜寻以找到针对包括目标地址的区域的条目，这可能需要访问多于一个条目以将目标地址与该条目的区域标识参数进行比较。因此，如果目标地址位于该表的给定存储器保护条目的对应地址区域中，则可以发出至少一个MPU存储器访问请求，并且所发出的(一个或多个)MPU存储器访问请求中的至少一者可以将给定的存储器保护条目作为目标(并且通常还会存在针对其他存储器保护条目的MPU存储器访问请求)。

[0049] 对于每个存储器保护条目，MPU可以允许对应地址区域的起始地址和结束地址不与N字节地址边界对准，其中N字节是对应地址区域的大小。这使得MPU能够管理在未对准的地址处开始的任何任意大小的单个地址区域的访问权限，这有助于避免将具有共享性质的区域人为分割成多个条目，这在将区域约束为在对准的地址边界处开始的系统中是需要的，基于上文讨论的理由，这可以有助于改进实时代码的性能。

[0050] MPU可以具有静态MPU寄存存储装置以用于存储至少一个静态存储器保护条目。该至少一个静态存储器保护条目可以提供具有指定的访问权限的地址区域的进一步定义，以补充存储在存储器系统中的存储器保护表的存储器保护条目中所定义的权限。提供一些配置选项以提供静态寄存存储装置中所定义的额外MPU区域可以用来将中断延迟保持较低并且提供了确定性中断行为。例如，静态MPU寄存存储装置可为要由性能关键(performance-critical)的代码访问的某些关键地址区域定义访问权限，以便与在这些访问权限需要由从存储器系统加载的存储器保护条目检查的情况相比，提供更快速的响应。因此，在具有静态MPU寄存存储装置的实施方式中，MPU实际上可以是混合式MPU，该混合式MPU使用定义一些地址区域的基于寄存器的条目和定义其他地址区域的基于存储器的条目两者。这可以在中断延迟与某些系统所需的实时保证之间、和在实现MPU的硬件成本与由软件(其需要在给定时间定义较大总数的存储器保护条目)实现的性能之间提供良好的平衡。

[0051] 静态MPU寄存存储装置就以下意义而言是静态的：在没有系统重置或断电事件的情况下，一旦被由处理电路处理的至少一个MPU参数配置操作所配置，存储在静态MPU寄存存储装置中的静态存储器保护条目就保持存储在静态MPU寄存存储装置中，直到处理电路处理另外的MPU参数配置操作以触发静态存储器保护条目的重新配置。MPU参数配置操作可以是改变针对给定地址区域指定的属性的任何操作(例如改变访问权限或改变区域的界限地址)，使得针对地址空间的一部分处理存储器访问的方式能够改变。MPU参数配置操作可以是由处理电路执行的专用MPU配置指令，或可以是下述存储指令：该存储指令将映射到MPU的静态MPU寄存存储装置的地址指定为该存储指令的目标地址。MPU参数配置操作也可以是由调试器或在处理电路外部的其他实体所请求的由处理电路执行的操作。因此，在静态存储器保护条目在静态MPU寄存存储装置中是活跃的(active)时候，这与缓存的不同之处在于，无论哪些地址被系统访问，静态存储器保护条目都将保留，直到它被重新配置为止。相比之下，在不需要由处理电路执行任何MPU配置操作的情况下，缓存会将该缓存的条目在该条目在一段时间内未被使用时或基于一些其他的逐出策略驱逐出，以便为用于可能需要的不同地址的条目腾出位置。在缓存的情况下，虽然缓存的特定条目可随时间改变，但这并不意味着，由存储器保护表的给定存储器保护条目指定的底层属性有任何改变，因为缓存占用的改变仅影响可获得给定存储器保护条目的延迟，并不改变对给定地址的存储器

访问的处理方式的功能定义。

[0052] 响应于处理电路发出的存储器访问请求,在目标地址是在存储在静态MPU寄存存储装置中的静态存储器保护条目的对应地址区域内、并且所发出的存储器访问请求满足由该静态存储器保护条目指定的访问权限时,MPU存储器访问电路可以抑制至少一个MPU存储器访问请求被发出到存储器系统。因此,在静态存储器保护条目中存在命中时,不需要触发用于取出存储器保护表的存储器保护条目的任何MPU存储器访问请求。这有助于预留可以用于其他操作的存储器带宽,并且意味着,使用存储在静态MPU寄存存储装置中的静态条目的性能关键软件不会由于基于存储器的区域查找而暂停。实际上,静态存储器保护条目可以比存储器保护表的基于存储器的条目优先。

[0053] 可以用不同的方式来处理存储器访问请求的目标地址命中静态存储器保护条目但访问请求无法满足由该静态存储器保护条目指定的访问权限时的情况。在一些情况下,在存储器访问请求无法满足静态存储器保护条目(针对该静态存储器保护条目,目标地址在对应的地址区域中)中指定的访问权限时,也可以抑制至少一个MPU存储器访问请求。因此,在此情况下,如果在静态MPU寄存存储装置中存在任何命中,则不需要访问存储器系统中的存储器保护表。

[0054] 替代地,对于命中静态存储器保护条目但未通过该静态存储器保护条目中的访问权限的存储器访问请求而言,可以允许该访问权限由存储在存储器系统中的存储器保护表内的匹配存储器保护条目(该匹配存储器保护条目包括下述条目:针对该条目的对应地址区域包括目标地址)指定的访问权限超控(override)。因此,静态MPU寄存存储装置中的命中在请求被允许时超控基于存储器的存储器保护条目中的任何权限组,但静态MPU寄存存储装置中的命中在请求不被静态条目允许时可以被存储器保护表的匹配存储器保护条目中指定的其他访问权限超控。此方法可以用来减少需要定义以允许第一软件进程对由第二软件进程访问的地址区域的特定子区域的访问的区域总数。例如,可以针对较广的地址区域设置静态MPU条目,以允许第二软件进程访问该区域但拒绝第一软件进程的访问。接着可以仅针对允许由第一软件访问的子范围指定涵盖较广区域内的较小地址区域的基于存储器的存储器保护条目。这可以避免需要针对在由第一软件访问的子范围的任一侧的较广范围部分提供单独的条目,从而为其他用途预留其他的静态存储器保护条目或实现存储器保护表的总大小的下降。

[0055] 每个静态存储器保护条目可以与未使用指示相关联,该未使用指示指示静态存储器保护条目是否为未使用的静态存储器保护条目。在给定的静态存储器保护条目被指示为是未使用的静态存储器保护条目时,MPU可以将来自存储器保护表的存储器保护条目缓存在静态MPU寄存存储装置的与该给定静态存储器保护条目相对应的部分中。因此,在软件不需要与硬件中所支持的一样多的静态存储器保护条目时,则可以将硬件寄存存储装置的一部分重用为缓存以实现对来自存储器保护表的存储器保护条目的本地存储,使得与在需要从存储器系统取出条目的情况相比,可以更快速地访问条目。这可以改进性能。在给定的静态存储器保护条目被标记为未使用时,哪个特定的存储器保护条目被缓存在静态MPU寄存存储装置中可根据缓存替换策略而随时间不同,即使在处理电路还未执行任何MPU参数配置操作时也如此。

[0056] 因此,在静态存储器保护条目被标记为已使用(即,该条目不具有设置的未使用指

示)时,则静态MPU寄存存储装置的该部分中的数据可以保持被存储,直到被MPU参数配置操作重新配置为止,但在静态存储器保护条目被指示为未使用时,则该静态存储器保护条目可以被用作缓存,在这种情况下,可以根据如由处理电路所执行的地址访问模式来改变静态MPU寄存存储装置中提供的特定条目。对于未被标记为未使用的静态存储器保护条目,还可以存在单独的启用/停用指示,该启用/停用指示在被设置为启用时意味着条目是有效的并且可以被存储器访问请求的目标地址命中,而对于停用的条目,即使目标地址匹配由停用的条目定义的区域,MPU也会总是返回未命中。停用的条目与未使用的条目的不同之处可在于,静态MPU寄存存储装置的与停用静态存储器保护条目对应的部分不能被用作缓存来自存储器中的存储器保护表的条目的缓存。软件可以暂时停用特定的静态存储器保护条目,使得对该地址范围的访问被拒绝,但继续将静态存储器保护条目保持在静态MPU寄存存储装置内,使得如果必要的话之后可以重新启用该静态存储器保护条目,而在未使用的静态存储器保护条目的情况下则意味着,可以将对应的寄存存储装置用作缓存来改进性能,但不保证在设置未使用指示之前的静态MPU寄存存储装置的内容能够被保留。

[0057] MPU可以包括MPU缓存存储装置,用以缓存保护表的至少一个存储器保护条目。在条目如上文所讨论地被标记为未使用时的情况下,MPU缓存存储装置可以是静态MPU寄存存储装置。然而,提供被提供以缓存来自存储器保护表的条目的专用MPU缓存存储装置也可以是有用的,该专用MPU缓存存储装置充当缓存,无论已经如何配置了静态存储器保护条目(若提供)。可以使用MPU内的与静态MPU寄存存储装置类似的另外一组寄存器来实现MPU缓存存储装置,或可以使用不同类型的数据存储装置(例如SRAM)来实现MPU缓存存储装置。在一些情况下,静态MPU寄存存储装置和MPU缓存存储装置可以是同一存储结构的部分,其中一些部分被分割以用作缓存存储装置,并且其他部分被分割以存储静态配置的存储器保护条目。

[0058] 因此,MPU缓存存储装置被提供以缓存存储器保护表的至少一个存储器保护条目。MPU存储器访问电路可以使用缓存替换策略(例如最近最少使用、依次轮流(round robin)等等)来确定在给定的时间将来自存储器保护表的哪一组存储器保护条目缓存在MPU缓存存储装置中。这意味着,与静态MPU寄存存储装置的存储未被标记为未使用的静态存储器保护条目的部分相比,即使处理电路还未执行任何MPU参数配置操作,也可以更新MPU缓存存储装置。

[0059] 响应于处理电路发出的存储器访问请求,MPU存储器访问电路可以根据目标地址是否在针对存储在MPU缓存中的任何存储器保护条目指定的对应地址区域内,来确定是否发起至少一个MPU存储器访问请求。例如,首先可以查找MPU缓存存储装置,并且如果在MPU缓存存储装置中存在命中,则可以不需要发起任何MPU存储器访问请求。缓存替换策略可以旨在保留被预测为在将来最有用的存储器保护条目,以便减少存储器访问请求中需要发出MPU存储器访问请求的部分。因此,如果在MPU缓存存储装置中存在命中,则可以抑制至少一个MPU存储器访问请求。

[0060] 处理电路可以响应于条目标识符查询指令来取回条目标识符,该条目标识符标识存储器保护表中获得给定的存储器保护条目的位置。例如,条目标识符查询指令可以指定目标地址,并且作为响应,处理电路和/或MPU可以返回对匹配该目标地址的存储器保护条目的条目标识符的指示。可以通过将所取回的条目标识符写入到处理电路的架构寄存器,

使得条目标识符可由执行在处理电路上的软件访问。MPU缓存存储装置可以针对每个缓存的存储器保护条目指定与存储器保护表中的获得该缓存的存储器保护条目的位置相关联的条目标识符。这意味着,对于缓存的存储器保护条目,处理电路可以更快速地服务条目标识符查询指令,因为无需向存储器系统发起MPU存储器访问请求以标识与在缓存的存储器保护条目中命中的地址相关联的条目标识符。能够取回给定存储器保护条目的条目标识符(而非仅返回访问权限)的指令可以用于加速某些类型的安全检查或提供代码段将以确定性时间运行的置信度(confidence)。通过针对给定的地址对执行条目标识查询指令,接着如果两个条目标识符查询指令都返回相同的条目标识符,则可以确定,地址对之间的整个地址范围对应于同一存储器保护条目,使得可以保证将不会有由于访问所指定的地址对之间的未映射的地址区域引起的错误。这同样可以改进性能,因为这意味着,在已经确定整个地址范围完全落在单个存储器保护区域内时,可以针对该范围执行单次访问权限检查,而不是需要单独检查范围内的每个单独地址。因此,除了缓存的存储器保护条目本身的任何参数以外还将与该条目相关联的条目标识符缓存在MPU缓存存储装置中可以在执行这样的条目标识符查询指令时用于加速处理电路的执行。

[0061] 处理电路可允许多个安全状态下的操作,每个状态可被细分成多个特权等级。处理电路可以允许安全状态之间的快速转换,例如通过允许分支到其他的安全状态的代码。多个安全状态中的每一者可以具有要在处理电路在该安全状态下执行指令时使用的相关联的存储器保护表。为了减少所需的MPU缓存存储量,可以在多个安全状态之间共享MPU缓存存储装置,其中缓存内的每个条目指定了安全状态标识符,其用于在从存储器保护表获得保护条目时标识处理电路所使用的安全状态。MPU可以抑制存储在MPU缓存存储装置中的下述缓存的条目的使用:针对这些缓存的条目,安全状态标识符标识的安全状态与处理电路当前活跃的安全状态不同。因此,可以将与非活跃安全状态相关联的条目保持缓存在MPU缓存存储装置中,但在当前活跃安全状态表与由安全状态标识符所标识的安全状态不同时不被检测为匹配目标地址。这在不同的安全状态之间切换时实现了改进的性能,因为即使活跃安全状态被处理电路重新配置,如果活跃安全状态之后被切换回先前活跃安全状态,则与该安全状态相关联的一些缓存的存储器保护条目仍然会存在于缓存中并且因此可以被更快速地访问以改进性能。

[0062] 替代地,取代使用安全状态标识符标记缓存的存储器保护条目,另一种方法可以是响应于活跃安全状态被处理电路修改,而使MPU缓存存储装置中的所有条目无效。虽然这可能需要在活跃安全状态被切换之后用来自与新活跃的安全状态相关联的存储器保护表的条目来重新填充缓存,但此方法可以使得MPU缓存存储装置更小,因为不需要为每个缓存的存储器保护条目提供安全状态标识符字段。

[0063] MPU缓存存储装置可以为每个缓存的保护条目指定与从中获得该缓存的存储器保护条目的存储器保护表相关联的表标识符。可以在存储器系统内建立多个单独的存储器保护表,例如对应于不同的软件部分的不同表。每个存储器保护表可以通过给定表标识符来标识。例如,表标识符可以是指派给该表的任意数值,或可以将对存储器保护表的基础地址的指示用作表标识符。该装置可以具有用于存储活跃表标识符的活跃表标识符寄存器,该活跃表标识符标识了当前将用来检查对存储器的访问权限的活跃存储器保护表。MPU可以抑制存储在MPU缓存存储装置中的下述缓存的存储器保护条目的使用:对于该缓存的存储

器保护条目，与该缓存的存储器保护条目相关联的表标识符与存储在活跃表标识符寄存器中的活跃表标识符不同。因此，可以将与非活跃存储器保护表相关联的条目保持缓存在MPU缓存存储装置中，但在当前活跃的表是不同的存储器保护表时不被检测为匹配目标地址。这在使用不同存储器保护表的不同软件进程之间切换时允许改进性能，因为即使活跃表标识符寄存器被处理电路重新配置为切换到不同的存储器保护表，如果活跃存储器保护表之后被切换回先前访问的表，则与该表相关联的一些缓存的存储器保护条目仍然会存在于MPU缓存存储装置中并且因此可以被更快速地访问以改进性能。

[0064] 替代地，取代使用对应的表标识符来标记缓存的存储器保护条目，另一种方法可以是响应于活跃表标识符寄存器被处理电路修改，而使MPU缓存存储装置中的所有条目被。虽然这可能需要在活跃表被切换之后用来自新活跃的存储器保护表的条目来重新填充缓存，但该方法可以使得MPU缓存存储装置更小，因为不需要针对每个缓存的存储器保护条目提供表标识符字段。

[0065] 响应于存储器保护缓存无效操作，处理电路可以触发MPU使存储在MPU缓存存储装置中的满足无效条件的一个或更多个缓存的存储器保护条目无效。例如，存储器保护缓存无效操作可以是由处理电路执行的特定指令。存储器保护缓存无效操作可以指定用于无效条件的许多不同准则中的一者。例如，第一类型的缓存无效操作可以指定应使MPU缓存存储装置中的所有条目无效。第二类型的存储器保护缓存无效操作可以标识要被无效的缓存的存储器保护条目的特定地址或地址范围，其中缓存的存储器保护条目定义了并非部分地或完全地与所预留的指定地址范围重叠的区域。用于无效条件的其他选项可以是，如果缓存的存储器保护条目与特定的表标识符、安全状态标识符中的至少一者对应、或具有特定范围内的条目标识符，则应使这些缓存的存储器保护条目无效。因此，存在许多不同选项用于控制对来自缓存的条目的无效。这是有用的，因为处理电路在软件的控制之下重新配置存储器保护表的一部分以改变针对特定地址区域所指定的参数时，则也可以触发对来自MPU缓存存储装置的对应条目的无效，以确保不将访问权限的旧值保留在缓存中。

[0066] 存储在存储器系统中的存储器保护表中的存储器保护条目可以包括对存储器保护条目是提供有效访问权限的启用条目还是提供无效访问权限的停用条目的指示。这可以用于暂时停用对某些地址区域的访问，同时保留停用的存储器保护条目中的区域参数，使得一旦该区域之后被启用，就不需要重新配置这些区域参数。MPU缓存存储装置可被禁止缓存存储器保护表的停用条目。例如，MPU存储器访问电路在查找表时可以仅针对启用条目返回命中信号，并且可以仅在存储器中的存储器保护表中检测到命中时将存储器保护条目分配给MPU缓存存储装置。这可用于避免缓存来自存储器系统的停用条目，因为这意味着，可以将伪条目(dummy entry)插入到存储器内的存储器保护表中以供之后潜在的使用，并且可以在不需要无效任何缓存的条目的情况下配置和启用这些条目。

[0067] 可以用不同的方式表示活跃存储器保护表的表标识符。在一个示例中，可以单独使用基础地址来标识表标识符。替代地，在一些情况下，可以使用基础地址和大小的组合来标识表。如果要由两个不同的软件进程访问的存储器保护表部分重叠，则这可以是有用的，使得可以将存储器保护表的共享部分存储在起始于基础地址处的地址处，然后可以将存储器保护表的仅将用于一个特定软件进程的部分定位在地址空间的后续部分处，使得可以简单通过将大小指示改变为指示活跃存储器保护表有多大，来实现在由不同软件部分访问的

不同存储器保护表之间的切换。在一些情况下，可以将基础地址和大小指定在MPU的单个寄存器中。然而，在支持相对较大的地址的系统中，基础地址和大小指示可能不配合于一个寄存器，并且因此有时候可能需要单独的寄存器。因此，可以提供基础地址寄存器来存储响应于处理电路发出的存储器访问请求而要由MPU存储器访问电路访问的活跃存储器保护表的基础地址，并且可以提供大小寄存器来存储对活跃存储器保护表的大小的指示。应理解，基础地址寄存器和大小寄存器两者也可以分别除了基础地址和大小以外还指定其他参数。

[0068] 在活跃存储器保护表的基础地址和大小被指定在不同的寄存器中时，则这意味着，可能需要由处理电路执行单独的MPU配置操作以分别更新活跃存储器保护表的基础地址和大小。这可能存在竞争情况的风险，其中未定义的行为发生在用来更新基础地址和大小中的一者的第一配置操作与用来更新基础地址和大小中的另一者的第二配置操作之间的时段中，因为改变基础地址和大小中的一者但不改变另一者可能存在区域被指示为包括旨在为存储器保护条目的信息的活跃表的风险。

[0069] 为了减少此风险，响应于指定要对活跃存储器保护表的基础地址和大小中的一者进行更新的预定类型的MPU配置操作，除了更新基础地址和大小中的所述一者以外，处理电路还会将基础地址和大小中的另一者更新为指示不存在活跃存储器保护表的无效值。通过在基础地址和大小中的一者已经被更新时有效地完全停用对存储器保护表的访问，直到另一者也已经被更新为止，则不会存在由下述MPU存储器访问请求所造成的任何不正确的操作：该MPU存储器访问请求访问地址空间的不旨在为存储器保护表的一部分的部分。

[0070] 在一个实施方式中，预定类型的MPU配置操作可以是指定要对基础地址进行更新的操作，并且可以指定要将大小清除(clear)为指示不存在活跃存储器保护表的无效值。因此，基础地址和大小中被指定为要由预定类型的MPU配置操作更新的一者可以是基础地址。这更容易实现。预定类型的MPU配置操作可以是专用指令或命令，但也可以简单地是下述由处理电路执行的常规存储指令：该常规存储指令将映射到基础地址寄存器或大小寄存器中指定基础地址和大小中将不被设置为无效值的一者的一者的地址指定为该常规存储指令的目标地址。在一些情况下，更新基础地址或大小的所有MPU配置操作也可以将另一者清除为无效值。替代地，可能不总是需要这样，并且只有某些类型的MPU配置操作可以将基础地址和大小中的另一者清除为无效值。例如，用于触发MPU配置操作的特定类型的指令可以对基础地址和大小中不被该操作配置为指定值的另一者是否也应被清除为无效进行信号告知。在其他的示例中，可以通过基础地址寄存器本身的位来激活“将大小清除为无效的”控制，使得如果基础地址寄存器被写入并且“将大小清除为无效的”旗标被设置在要被写入到基础地址寄存器的新值中，则MPU也将把大小清除为无效值。

[0071] 由MPU所发出以从存储器保护表获得条目的MPU存储器访问请求本身可能需要针对某些访问权限进行检查。然而，可能不期望基于存储器检测表本身来检查MPU存储器访问请求是否被允许，因为这可能导致对存储器保护表的递归查找组，该递归查找组可能导致潜在非确定性的响应时间或造成死锁的永不结束的请求循环。可以使用不同的方法来处理对MPU存储器访问请求本身的检查。在一个示例中，权限检查电路可以基于独立于存储器保护表指定的访问权限预设组来确定至少一个MPU存储器访问请求是否被允许。因此，通过使用权限预设组(该权限预设组例如可以是硬连线的或在MPU的可配置寄存器中指定的)，不需要查找存储器中的存储器保护表来确定是否允许MPU向特定地址发出存储器访问请求。

这可以有助于改进实时处理。

[0072] 替代地,或除了对照访问权限预设组来检查MPU存储器访问请求以外,权限检查电路还可以基于至少一个MPU存储器访问请求的地址是否是在由多个存储器安全配置寄存器指定的多个地址范围中的至少一者中,来确定MPU存储器访问请求是否被允许。例如,存储器安全配置寄存器可以指定定义了哪些地址范围可以从上文讨论的多个安全状态中的给定一者被访问的信息。例如,一些系统可以支持定义地址空间中仅可以由某些安全进程访问的安全部分和地址空间中可以由安全和非安全的进程中的任一者访问的非安全部分,并且因此需要对照由存储器安全配置寄存器定义的安全/非安全区域来检查MPU存储器访问请求。在此情况下,如果由非安全代码触发的MPU存储器访问请求的目标是在存储器安全配置寄存器中被定义为安全地址空间的一部分的区域,则可以拒绝这些MPU存储器访问请求。由存储器安全配置寄存器定义的权限可以与由存储器保护表和静态MPU条目提供的权限正交(orthogonal)。因此,除了对照存储器安全配置寄存器来检查MPU存储器访问请求以外,也可以对照存储器安全配置寄存器来检查处理电路响应于指令取出请求或常规加载/存储指令而发出的存储器访问请求,以检查这些存储器访问请求是否被允许。

[0073] 在一些示例中,可以通过设置与缓存的存储器保护条目相关联的锁定指示来将该条目标记为“锁定”。MPU可以防止锁定的缓存的存储器保护条目被驱逐出缓存。例如,MPU可以防止与包括堆栈数据结构(用于响应于异常而保存架构状态并且响应于异常返回而恢复所保存的状态)的地址区域对应的锁定的缓存的存储器保护条目。通过锁定与堆栈对应的条目使得该条目不能被驱逐出,这通过使状态保存/恢复的延迟减少而确保了快速的异常处理响应。MPU可以使用存储在处理电路的堆栈指针寄存器中的堆栈指针地址来确定哪个条目与堆栈对应。可以在任何时间进行与堆栈相关联的MPU条目的锁定。然而,在一个示例中,响应于在状态从堆栈恢复时的异常返回,可以锁定与当前堆栈指针地址相关联的MPU条目,使得如果另外的异常或中断在异常返回期间或刚好在异常返回后发生,则仍然可在MPU缓存存储装置中取得该堆栈的条目,并且因此可以针对另外的异常或中断更快速地保存状态。

[0074] 在其他的实施例中,可以通过在MPU缓存存储装置内具有仅被用于存储来自存储器保护表的与堆栈数据结构相关联的条目的专用条目,来避免与每个缓存的存储器保护条目相关联的锁定指示的使用。可以在任何时间填充此专用条目。然而,在一个示例中,响应于在状态从堆栈恢复时的异常返回,可以将与当前堆栈指针地址相关联的MPU条目填充到专用条目中。

[0075] 在一些实施方式中,在响应异常情况时,处理电路可以使用惰态保存。在异常情况发生时,处理电路可能需要将与在异常发生之前执行的出局进程相关联的一些架构状态保存到存储器系统,使得即使在异常情况之后执行的进程重写该架构状态,所保存的状态也被保留。可以在处理从处理异常返回时在之后恢复与出局进程相关联的状态。将来自架构寄存器的状态保存到存储器系统可以是相对缓慢的,并且因此为了减少中断处理延迟,一些系统可以响应于异常情况本身而仅将架构状态的第一部分保存到存储器系统。可以不直接响应于异常情况而保存架构状态的第二部分,但可以在之后仅在异常情况之后执行的后续软件进程实际访问架构状态的第二部分时保存该第二部分。例如,架构状态的第二部分可以是用于存储浮点值的浮点寄存器的内容,而架构状态的第一部分可以是整数寄存器的

内容。这是有用的,因为许多软件进程可能不使用任何浮点寄存器,并且因此如果在异常情况之后执行的后续软件进程永不访问浮点寄存器,则将保留与先前执行的进程相关联的状态,并且因此不需要针对这些浮点寄存器执行状态保存和恢复。

[0076] 虽然这样的惰态保存对于性能来说可能是有用的,但在MPU使用存储在存储器系统中的存储器保护表时,这可能造成一些复杂性。为了允许在存储器中使用一致的堆栈框架以保留架构状态的第一部分和第二部分(无论架构状态的第二部分实际上是否已经被保存到存储器),响应于异常情况,处理电路可以记录对第二部分状态保存地址区域的指示:如果后续的软件进程访问架构状态的第二部分,则架构状态的第二部分将被保存在该第二部分状态保存地址区域处。因此,在后续的软件进程发起对存储有架构状态的第二部分的寄存器的访问时,处理电路可以接着读取标识第二部分状态保存地址区域的所记录的地址并且接着将架构状态的第二部分保存到所标识的地址区域。通过在架构状态的第二部分还未被保存的情况下也为该第二部分预留空间,这意味着,堆栈框架的大小对于每次异常都是相同的,使得在异常嵌套时,无论是否实际上保存了架构状态的第二部分,后续的堆栈框架的位置都是相同的。这使得控制状态恢复简单得多。

[0077] 然而,由于可能需要对照MPU来检查用于将架构状态的第二部分保存到存储器系统的存储器访问本身,并且活跃存储器保护表常常会响应于异常情况而改变,用于检查在异常情况之前执行的进程是否已被允许将该进程的架构状态的第二部分保存到针对第二部分状态保存地址区域所指定的地址的相关访问权限可能不再可用。

[0078] 因此,响应于异常情况,MPU可以记录MPU检查信息,用于使MPU在后续软件进程的执行期间能够确定用于将架构状态的第二部分保存到存储器系统的存储器访问是否会满足由在异常情况发生时活跃的存储器保护表的存储器保护条目指定的访问权限。MPU检查信息可以包括许多不同类型的信息。在一些情况下,MPU检查信息可以包括在异常情况发生时活跃的存储器保护表的表标识符(例如基础地址和可选的大小),使得在后续的软件进程的执行期间,可以标识先前活跃的相关存储器保护表,并且可以访问适当的条目以检查在异常情况之前执行的进程是否已被允许将状态保存到针对第二部分状态保存地址区域所指定的地址。MPU检查信息还可以包括对下述情形的指示:第二部分状态保存地址区域的起始地址和结束地址是否对应于在异常情况发生时活跃的存储器保护表中相同存储器保护条目。在第二部分状态保存地址区域的起始地址和结束地址均对应于同一存储器保护条目时,则这意味着,单次的访问权限检查可能足以确定整个第二部分状态保存地址区域是被允许访问的,从而避免了对照在异常情况发生时活跃的存储器保护表的条目检查第二部分状态保存地址区域的每个单独部分的需要。在一些实施方式中,可以在异常情况发生时进行单次访问权限检查,并且保存检查结果,而在其他的实施例中,可以保存访问权限,使得即使存储器保护表已经改变,也可以在之后执行检查。因此,通过将关于在异常情况发生时活跃的存储器保护表的一些信息保存,这使得即使活跃的存储器保护表响应于异常情况而改变,惰态保存也能够起作用。

[0079] 可以提供相应的计算机程序用于控制主机数据处理装置以提供供指令执行的指令执行环境。该计算机程序可以包括在功能上与上文讨论的处理电路和MPU对应的处理程序逻辑和MPU程序逻辑。因此,可以用与在实际上包括上文讨论的处理电路和MPU的硬件上执行的程序中的存储器访问指令类似的方式来处理在由计算机程序模拟的指令执行环境

内执行的程序中的存储器访问指令。模拟器计算机程序可以向在模拟器计算机程序上执行的软件呈现与会由实际的硬件装置提供的指令环境类似的指令环境，即使可能不存在在执行模拟器计算机程序的主计算机中提供这些特征的任何实际硬件。可以通过提供程序逻辑(例如指令集或数据结构)来模拟上文讨论的硬件架构的功能，该程序逻辑使得通用主计算机能够以与会在实际具有这样的MPU的装置上实现的结果兼容的方式来执行旨在在具有上文讨论的MPU支持的装置上执行的代码。可以将用于控制主机数据处理装置的模拟器计算机程序存储在存储介质上。此存储介质可以是非暂态存储介质。

[0080] 表查找

[0081] 上文讨论的存储器保护表是存储在存储器系统中的控制表的一个示例，该控制表提供由处理组件用来执行给定操作的信息。在存储器保护表的情况下，处理组件可以是MPU，并且所执行的操作可以是确定给定的存储器访问是否被允许。这样的控制表的另一示例可以是定义地址空间的哪些区域是安全区域还是非安全区域的表，并且基于控制表来执行的操作可以是确定处理电路是否应在安全模式还是较不安全的模式下进行操作，其中在所处理的指令地址是在控制表中定义的安全地址区域中的一者内时选择安全模式。另一示例可以是针对由控制表表示的每个地址区域定义与该区域相关联的软件库的标识符的控制表，该标识符可被处理电路用来确定是否允许给定的软件库调用另一软件库。

[0082] 因此，在处理情形的范围中，提供具有许多条目的控制表是有用的，每个条目指定第一界限地址和第二界限地址与大小中的任一者，使得第一界限地址和第二界限地址/大小标识地址空间内可变大小的对应地址区域的界限。条目可按照第一界限地址的顺序被布置在控制表中。虽然针对每个条目支持可变大小的区域可以用于减少针对给定大小的地址区域表示某个性质集所需的条目数量(避免需要将较大的地址区域人为分割成许多单独的固定大小的条目)，但允许可变大小的条目可以使得对控制表的查找的实现更复杂。这是因为，在支持可变大小的区域时，表内与给定地址对应的位置可能不是固定的，并且因此实际上为了定位与给定地址对应的条目，可能需要通过表的条目进行一些搜寻，以将查询地址与定义对应地址区域的界限的参数(第一界限地址和第二界限地址与大小中的一者)进行比较。因为地址空间被稀疏地分配有控制表中的条目，使得可能存在这样的一些地址：这些地址不具有针对包括该地址区域所定义的任何对应的条目，所以人们能够预计到，控制表的搜寻将需要对照第一界限地址和用来定义每个地址区域的其他界限的第二界限地址与大小中的一者进行的比较，并且因此可能需要大量的比较。

[0083] 在下文讨论的技术中，提供了表查找电路以执行二分搜寻过程以标识控制表是否包括对应于查询地址的条目。二分搜寻过程包括许多搜寻窗口窄化步骤，用于将控制表的候选条目的当前搜寻窗口窄化为包括更少候选条目的更窄搜寻窗口。在给定的搜寻窗口窄化步骤中，表查找电路独立于当前搜寻窗口的所选择的候选条目的第二界限地址或大小，基于查询地址与当前搜寻窗口的所选择的候选条目的第一界限地址进行的比较来窄化当前搜寻窗口。在将当前搜寻窗口窄化为控制表的单个候选条目的最终搜寻窗口窄化步骤之后，表查找电路可以确定，在查询地址是在由该单个候选条目的第一界限地址和第二界限地址或大小定义的对应地址区域内时，该单个候选条目是与查询地址对应的条目。

[0084] 因此，利用此方法，每个搜寻窗口窄化步骤仅需要将查询地址与控制表的给定选定候选条目的第一界限地址进行比较，所以不需要从存储器系统加载第二界限地址或大

小。仅需要针对由最终的搜寻窗口窄化步骤所得到的单个候选条目比较第二界限地址或大小。因此，对于控制表中给定数量的条目N，此方法允许比较次数与log2(N)+1对应而不是与log2(N)*2对应，与后者对应是需要针对每个搜寻窗口窄化步骤检查第二界限地址或大小时的情况。这可以表示所需的比较次数和需要从存储器系统加载以获得在二分搜寻过程中搜寻的候选条目的相应界限的数据量的显著减少。因此，这可以提供使得能够以较佳的性能查找存储在存储器中的控制表的查找程序，从而加速了对控制表的访问并且因此加速了依赖来自控制表的信息的操作的性能。

[0085] 在二分搜寻过程中，在最终的搜寻窗口窄化步骤之后标识的单个候选条目可以是控制表中唯一的下述条目：针对该条目，表查找电路触发存储器访问请求以便从存储器系统获得第二界限地址或大小，来作为针对查询地址执行二分搜寻过程的结果。因此，不需要针对控制表的该单个候选条目以外的任何条目实现任何加载以便从存储器系统获得第二界限地址或大小。这减少了所需的加载次数并且因此改进了性能。请注意，并不排除这样的情况：在执行二分搜寻过程以便基于第一查询地址查找表的同时，作为查找与第一查询地址不同的第二查询地址是否在表中具有对应条目的另外的查找的一部分，可能仍然存在被执行以从存储器获得该单个候选条目以外的候选条目的第二界限地址或大小的加载。可以针对不同的查询地址并行处理对表的多个查找，在这种情况下，第二查找可能仍然需要在针对第一查询地址的查找中标识的单个候选条目以外的候选条目的第二界限地址或大小。因此，限制仅针对在最终的搜寻窗口窄化步骤之后标识的单个候选条目加载第二界限地址或大小仅适用于由于针对当前查询地址执行二分搜寻过程而触发的存储器访问请求组，并且并不排除可能由于正与当前查询地址的二分搜寻过程并行执行的其他操作而触发的用于获得第二界限地址或大小的对表中其他条目的加载。

[0086] 在一些实施方式中，在表查找电路与包含控制表的存储器之间在存储器系统中可存在缓存。表查找电路向缓存作出请求，并且如果请求不能被缓存满足，则缓存可以作为响应而对存储器作出请求以取回所请求的数据。在一些实施例中，缓存可以请求比原本请求的数据更多的数据，例如它可以请求足以填充整个缓存线的数据。因此，在表查找电路仅针对第一界限地址对缓存执行请求时，缓存可以针对包括第一界限地址和第二界限地址或大小两者的缓存线对存储器执行请求。应理解，即使缓存正在请求第一界限地址和第二界限地址或大小两者，表查找电路也可能需要接收第一界限地址并对照第一界限地址进行比较(除了在最终搜寻窗口窄化步骤之后标识的单个候选条目以外)，并且因此降低了所使用的功率和电路复杂性。

[0087] 在查询地址是在由在最终的搜寻窗口窄化步骤之后标识的单个候选条目的第一界限地址和第二界限地址或大小定义的对应地址区域之外时，则表查找电路可以确定，查询地址在控制表中未命中。也就是说，可以确定，控制表并不包括与包括查询地址的地址区域相关的任何条目。在查询地址在控制表未命中时，则表查找电路可以执行以下步骤中的至少一者：对异常情况进行信号告知，并且返回预设信息以便控制要由处理组件所执行的操作。

[0088] 可以用不同的方式来实现在每个搜寻窗口窄化步骤处执行的比较。在一些示例中，比较可以是查询地址与表示对应地址区域的下界限的下界限地址的小于比较。因此，在此情况下，第一界限地址可以包括对应地址区域的起始地址。在比较每个区域的起始地址

时，则在该比较标识出查询地址小于所选择的候选条目的起始地址时，则可以将搜寻窗口窄化为使得从搜寻窗口排除控制表中具有大于或等于所选择的候选条目的起始地址的起始地址的至少一个条目。

[0089] 替代地，在每个搜寻窗口窄化步骤处的比较可以将标识对应地址区域的上界限的结束地址与查询地址进行比较。在此情况下，比较可以是大于比较。在此情况下，在查询地址大于所选择的候选条目的上界限地址(结束地址)时，可以将当前搜寻窗口窄化为从更窄的搜寻窗口排除控制表中具有等于或小于所选择的候选条目的结束地址的结束地址的至少一个条目。

[0090] 可以选择不同的选项用于确定所选择的候选条目本身在给定的搜寻窗口窄化步骤之后是否仍然是更窄搜寻窗口的一部分。在第一界限地址是起始地址并且比较确定了查询地址大于或等于起始地址时，则可能的情况是，查询地址可能在所选择的候选条目的对应地址区域内，并且因此所选择的候选条目在该搜寻窗口窄化步骤之后可以仍然是更窄搜寻窗口的一部分。类似地，如果第一界限地址是结束地址并且比较指示查询地址小于或等于所选择的候选条目的结束地址，则再次地，所选择的候选条目可以仍然是更窄搜寻窗口的一部分。

[0091] 在一些实施方式中，在给定的搜寻窗口窄化步骤在比较中确定了查询地址是在所选择的候选条目的对应地址区域之外时(例如因为查询地址小于所选择的候选条目的起始地址或大于结束地址)，则对于一些实施方式而言，可以从由给定的搜寻窗口窄化步骤所得到的更窄搜寻窗口排除所选择的候选条目。也就是说，因为已经知道所选择的候选条目不会是与查询地址对应的条目，可能需要从窄的搜寻窗口排除所选择的候选条目，使得后续的搜寻窗口窄化步骤可以更快速地标识出与查询地址对应的特定条目，因为在其余的搜寻窗口窄化步骤中需要考虑更少的条目。

[0092] 然而，在其他的实施方式中，在已经知道查询地址是在所选择的候选条目的对应地址区域之外时，所选择的候选条目仍然可以保持被包括在更窄的搜寻窗口中。这似乎违反直觉，因为人们可能不会想到在更窄的搜寻窗口内包括已知不与查询地址对应的候选条目是有用的。然而，实际上，无论比较结果如何都将所选择的候选条目包括在更窄的搜寻窗口内可以允许具有减少的关键路径时序的更高效的电路实现。例如，可以在一些实施方式中使用相应的条目标识符来追踪当前搜寻窗口的界限和所选择的候选条目的位置。因此，如果要从更窄的搜寻窗口排除所选择的候选条目，则可能需要通过将所选择的候选条目的标识符加一或减一，来从所选择的候选条目标识符计算要用于下个搜寻窗口窄化步骤的更窄搜寻窗口的起始和结束。该±1操作可能需要额外的硬件来加1或减1并且可能是在关键时序路径上。因此，在一些实施方式中，如果无论比较结果如何都仍然将所选择的候选条目包括在窄的搜寻窗口中，则标识条目以包括在窄的搜寻窗口中是更快速的。实际上，与所选择的候选条目的选定情况被排除时的情况相比，这不大可能造成需要任何额外的搜寻窗口窄化步骤，因为实际上，针对窄化的搜寻窗口的不同候选条目执行比较的后续搜寻窗口窄化步骤可能之后的结果无论如何都会排除先前的搜寻窗口窄化步骤中所标识的选定候选条目。因此，在一些实施方式中，通过提供将更窄搜寻窗口设置为在所选择的候选条目处的逻辑起始或结束，这可以允许具有较短的关键时序路径的更高效的控制逻辑。

[0093] 给定的搜寻窗口窄化步骤内所选择的候选条目可以是当前搜寻窗口的任何条目

(在一些情况下包括搜寻窗口的外边界中的一者处的条目)。然而,如果所选择的候选条目是针对该搜寻窗口窄化步骤所提供的当前搜寻窗口的中间点处的条目,则二分搜寻过程可能更高效。中间点不需要是在当前搜寻窗口中所包括的条目的准确中点处。事实上,在当前搜寻窗口包括偶数个条目时,则不可能选定准确中点处的条目,因为无论将哪个条目选择为所选择的候选条目,该条目都将比另一端更靠近搜寻窗口的一端。如果在每个搜寻窗口窄化步骤处,所选择的候选条目尽量靠近当前搜寻窗口的中点,则可以减少表中给定数量的条目所需的总搜寻窗口窄化步骤数的最大限值。然而,这不是必需的,并且在一些实施方式中,可以使用当前搜寻窗口的中间点处的任何条目,这可以导致功率和电路复杂性的降低。

[0094] 请注意,在搜寻窗口窄化步骤中,在搜寻窗口仅包括2个候选条目时,将2个条目中的一者选择为所选择的候选条目,即所选择的候选条目将在搜寻窗口的外边界中的一者处。因此,在一些情况下,仅对于当前搜寻窗口包括至少3个候选条目的搜寻窗口窄化步骤而言,所选择的候选条目可以在当前搜寻窗口的中间点处。

[0095] 在一些实施方式中,该装置可以具有存储至少一个条目的条目存储装置。条目存储装置可以是条目缓存,用于缓存先前从存储器系统获得的控制表的至少一个条目,和/或是静态寄存存储装置,用于存储可由静态条目参数配置操作配置的至少一个另外的条目(其中每个另外的条目指定地址空间内可变大小的对应地址区域的控制信息,对于该对应地址区域,界限通过第一界限地址和第二界限地址与大小中的一者来标识)。例如,在控制表是存储器保护表的情况下,条目缓存可以对应于上文讨论的MPU缓存存储装置,并且静态寄存存储装置可以对应于上文讨论的静态MPU寄存存储装置。因此,查找电路除了有权访问存储在存储器系统中的控制表以外还可以有权访问本地存储的一些条目,以提供对一些条目的更快速访问。

[0096] 如果至少一个控制条目被存储在条目存储装置中,则表查找电路可以检查查询地址是否与存储在条目存储装置中的至少一个条目对应,并且根据查询地址是否与存储在条目存储装置中的任何条目对应,来确定是否执行二分搜寻过程。例如,在查询地址与存储在条目存储装置中的条目对应时的情况下,可以完全省略二分搜寻过程。例如,如果所需的条目已经被缓存在条目缓存中,则可能不需要执行对存储器的任何访问,并且因此可以省略二分搜寻过程。对于静态寄存存储装置中的条目,如上文针对MPU示例所讨论,不同的选项可以控制在静态寄存存储装置中存在命中时的情况下是否需要执行二分搜寻过程。如果在静态寄存存储装置中存在任何命中,则在一些情况下可以抑制执行二分搜寻过程。替代地,其他示例可以考虑静态寄存存储装置中的匹配条目内指定的特定参数,以确定是否执行二分搜寻过程(例如如上文通过利用MPU示例所讨论,可以仅在匹配的静态配置的条目的参数指示存储器访问不被允许时才需要搜寻存储器内的表,但可以在指示存储器访问被允许的静态配置的条目中存在命中时省略二分搜寻过程)。

[0097] 在执行二分搜寻过程时的情况下,在一些实施方式中,表查找电路可以使用从存储在条目存储装置中的至少一个条目推导的信息来改进二分搜寻过程的效率。例如,即使在给定的查询地址在条目缓存或静态寄存存储装置的条目中未命中时,条目不匹配命中这些本地存储的条目所需的任何需要的比较的事实也可以提供下述信息:允许知道在二分搜寻过程内执行的一些比较的结果而不需要从存储器加载任何信息。例如,表查找电路可以

基于从存储在条目存储装置中的至少一个条目推导的信息，来限制用于二分搜寻过程的初始搜寻窗口窄化步骤的当前搜寻窗口。此外，在二分搜寻过程的给定搜寻步骤中，在查询地址与所选择的候选条目的第一界限地址的比较结果可从存储在条目存储装置中的至少一个条目中的信息推导时，可以抑制用于从存储器系统获得所选择的候选条目的第一界限地址的存储器访问请求。因此，通过使用已经可从本地存储的条目取得的信息来引导二分搜寻过程，这可以避免不必要的用于获得可能已经隐含在本地存储的信息中的信息的存储器访问，从而为二分搜寻过程提供更快速的性能。

[0098] 例如，在一个实施方式中，在针对给定的查询地址查找条目存储装置时，可以并行地查询条目存储装置中的所有条目，以确定哪些条目(并且因此确定表中的哪些索引)大于/小于查询地址。此时，用于第一搜寻窗口窄化步骤的初始搜寻窗口的下边界可以是小于查询地址的地址区域的最靠近条目的索引，并且初始搜寻窗口的上边界可以是大于查询地址的地址区域的最靠近条目的索引。替代地，允许二分搜寻过程用较小的搜寻窗口起始(可能加速二分搜寻)但需要一些额外的电路逻辑来加1或减1的方法可以是，将初始搜寻窗口的下边界设置为小于查询地址的地址区域的最靠近条目的索引加1；并且将上边界设置为小于查询地址的地址区域的最靠近条目的索引减1。

[0099] 可以将二分搜寻过程设计为使得每个搜寻窗口窄化步骤提供当前搜寻窗口的二分分割以获得更窄的搜寻窗口。例如，从给定的搜寻窗口窄化步骤所得到的更窄搜寻窗口可以是当前搜寻窗口的上部或下部中的任一者，其中搜寻窗口的上部和下部中的该者是基于查询地址与所选择的候选条目的第一界限地址的比较来选择的。

[0100] 可以将二分搜寻过程设计为处理控制表具有与2的幂对应的条目数N的情况。然而，有时候，控制表可以具有并非准确的2的幂的条目数N。在此情况下，表查找电路可以执行二分搜寻过程，其中假设初始搜寻窗口窄化步骤的当前搜寻窗口包括的候选条目数对应于大于N的最小的2的幂。在此情况下，可能的情况是，在特定的搜寻窗口窄化步骤中，当前搜寻窗口的所选择的候选条目与控制表的N个条目的范围之外的位置对应。如果这发生了，则表查找电路可以抑制用于从存储器系统获得所选择的候选条目的第一界限地址的存储器访问请求，并且由于要针对该特定的搜寻窗口窄化步骤执行比较，确定隐含结果，该隐含结果使得更窄的搜寻窗口包括控制表的N个条目的范围之内的位置处的至少一个候选条目。因此，如果控制表中的条目数不是2的幂，则可以以如同区域表是下个最大的2的幂的相同方式执行二分搜寻过程，但对于针对表的有效范围之外条目的比较，可以在实际上不从存储器加载值的情况下返回隐含结果。这使得控制表能够支持任何任意数量的条目，同时仍然使得能够通过对当前窗口大小执行除2来确定所选择的候选条目的位置(例如在中间点处)。因为可以通过位移位操作来执行除2，这可以减少电路复杂性。

[0101] 对于可变大小被允许是非2的幂的字节数和/或对应地址区域的起始地址和结束地址被允许不与X字节的地址边界(其中X是对应地址区域的大小)对准的控制表，上文讨论的二分搜寻过程可以是特别有用的。可能难以对具有这些性质中的一者的表加索引，因为不可能基于查询地址简单地将索引加到所需的条目中。取而代之，可能需要用于搜寻表的多个条目并比较界限地址的搜寻过程。上文讨论的技术使得能够更快速地搜寻这类表。

[0102] 在一些实施方式中，可以将第一界限地址存储在存储器系统内的与用来存储第二界限地址和大小中用来定义对应地址区域的另一界限的一者的数据字相同的数据字中。

“数据字”指的是在单个存储器访问请求中可以从存储器返回的一部分数据的大小。即使第一界限地址和第二界限地址或大小被存储在相同的数据字中，使用上文讨论的二分搜寻过程来避免针对每个搜寻窗口窄化步骤中检查的每个所选择的候选条目返回第二界限地址或大小的需要仍然是有用的。这是因为，通过在二分搜寻过程的每个搜寻窗口窄化步骤中仅需要第一界限地址，这意味着，可以使用指定更窄的目标数据部分的存储器访问请求，这可以允许更高效的存储器带宽使用，因为这可以保留存储器系统内的寄存器中的空间，或保留总线上未用于存储器访问请求的导线，这可以随后重新用于其他的存储器访问，从而改进存储器系统的整体效率。仅需要在搜寻窗口窄化步骤中对照第一界限地址进行比较也减少了所需的比较电路的量并且可以减少所消耗的能量。

[0103] 然而，在对于控制表中的给定条目，第一界限地址是被存储在存储器系统内的与用来存储给定条目的第二界限地址或大小的数据字不同的字中的示例中，该技术是特别有用的。在一些系统中，数据字的大小可能与第一界限地址或第二界限地址的大小相当，并且因此，在相同的数据字内存储第一界限地址和第二界限地址或大小两者可能是不实际的。在此情况下，如果需要在每个搜寻窗口窄化步骤处获得第一界限地址和第二界限地址或大小两者，则这对于每个搜寻窗口窄化步骤将需要两个单独的存储器访问请求，这会增加对存储器带宽的压力。相比之下，通过使用上文讨论的二分搜寻过程，可以避免用于访问每个所选择的候选条目的第二界限地址或大小的存储器请求，其中唯一需要从存储器加载第二界限地址或大小的条目是导致最终窗口窄化步骤的单个候选条目。

[0104] 虽然可以将上文讨论的技术用于如上文讨论的支持由第一界限地址和第二界限地址或大小定义的可变大小地址区域的任何类型的控制表，对于下述存储器保护表是特别有用的：针对该存储器保护表的每个条目指定地址空间的对应地址区域的访问权限，并且基于控制表中的信息来执行的操作包括检查指定查询地址的存储器访问请求是否满足包括查询地址的区域的对应条目中的访问权限。可能需要常常访问这样的存储器保护表(例如在每个存储器访问请求访问，如果所需的条目还未存储在本地的存储电路中的话)，并且因此改进用于从存储器访问存储器保护表的查找过程的效率可以对使用存储器保护表的处理系统的整体处理性能有很大的影响。

[0105] 再次地，与上文讨论的MPU示例类似，在一些情况下，可以使用硬件逻辑来实现该系统。然而，也可以在模拟器计算机程序内实现这样的二分搜寻过程，该模拟器计算机程序用于控制主机数据处理装置以提供供指令执行的指令执行环境，其中处理程序逻辑和表查找程序逻辑提供与上文讨论的处理组件和表查找电路相同的功能。存储介质可以存储模拟器计算机程序。存储介质可为非暂态存储介质。

[0106] 具体的MPU示例

[0107] 图1示意性地图示出数据处理系统2，该数据处理系统具有处理电路4，用于响应于指令而执行数据处理。例如，处理电路4可以是CPU、DSP、GPU、或数据处理系统的其他处理元件。处理电路4可以包括：指令取出单元5，用于从存储器系统10取出指令以供由该处理电路处理；和指令解码器7，用于将取出的指令解码以生成用于控制执行单元6、8以响应于解码的指令执行数据处理操作的控制信号。例如，执行单元可以包括算术/逻辑单元(ALU)6，用于对从寄存器12、14取出的操作数执行算术或逻辑运算以生成可以写回寄存器的结果值。执行单元也可以包括加载/存储单元8，用于执行加载/存储操作以便在寄存器12、14与存储

器系统10之间传输数据。寄存器可以包括许多不同类型的寄存器，包括：通用(整数)寄存器12，用于存储供处理电路4处理的整数值；和浮点寄存器14，用于存储使用浮点表示来表示的值。处理电路4可访问的寄存器还可以包括为了简洁起见未在图1中示出的其他类型的寄存器，例如：向量寄存器，用于存储包含多个独立数据元素的向量操作数；和/或控制寄存器，用于存储用于控制处理电路4的操作的各种控制参数，例如：条件旗标，其可以被条件指令检查以确定是否执行条件操作；程序计数器，其存储表示当前执行点的地址；以及控制寄存器，例如用于控制异常处理。

[0108] 存储器系统10可包括一个或更多个等级的缓存和主存储器。通过执行加载指令或存储指令而使存储器系统10可由处理电路4访问，该加载指令触发加载/存储单元8以发出存储器访问请求用于将来自存储器系统10的数据加载到寄存器12、14中，该存储指令用于触发加载/存储单元8以发出存储器访问请求用于将来自寄存器12、14的数据存储到存储器系统10中。此外，指令取出单元5发出存储器访问请求以从存储器系统10取出指令。每个存储器访问请求可以指定将要访问的位置的目标地址。在此示例中，目标地址是直接指定将要访问的位置的物理地址，所以不需要地址转换。

[0109] 提供了存储器保护单元(MPU)20用于基于目标地址检查是否允许存储器系统10为存储器访问请求服务。可以用两种方式定义用于控制是否允许处理电路4访问给定地址区域的访问权限。可以将存储器保护表22存储在存储器系统内，该存储器保护表包括许多条目，每个条目定义对应地址区域的访问权限。MPU 20具有MPU存储器访问电路24，用于向存储器系统10发起存储器访问请求以从MPU表22获得信息。MPU存储器访问电路24具有表查找电路26，用于执行标识MPU表22是否包括用于下述地址区域的条目所需的比较：该地址区域包括由从处理电路4接收的存储器访问请求指定的目标地址。存储器系统10可以存储多于一个MPU表，每个表22与相应的软件进程相关联。活跃表标识符(标识要使用MPU存储器访问电路24来访问哪个特定的MPU表)被存储在活跃表标识符寄存器(或寄存器组)28中。例如，活跃表标识符寄存器28可以指定当前活跃的MPU表的基础地址30，并且还可以可选地指定活跃表的大小。在MPU内提供了MPU缓存存储装置32，用于从存储器系统10缓存(一个或多个)MPU表的所选择的条目，使得如果再次需要相同的条目，则可以较快速地访问该条目。例如，MPU缓存存储装置可以包括一组寄存器，用于与针对存储器系统10中的MPU表22进行回存相比，以减少的访问延迟本地地存储MPU表的条目。

[0110] 提供访问权限来控制对存储器系统10的访问的另一种方式可以是在静态MPU寄存存储装置36内定义静态配置的存储器保护条目。处理电路4可以支持用于配置静态MPU寄存存储装置36的内容的指令(例如特别旨在访问静态MPU寄存存储装置的专用类型指令)，或可以使用由加载/存储单元8执行的更常规的存储操作，该存储操作将已经被映射到针对静态MPU寄存存储装置36提供的存储器映射的寄存器的地址指定为目标地址。通常而言，一旦已经执行MPU参数重新配置操作(例如这些存储器映射的加载/存储操作或专用类型的重新配置指令中的一者)以填充静态MPU寄存存储装置36的内容以定义与给定的地址区域对应的给定存储器保护条目，在没有造成存储在静态寄存存储装置36中的信息损失的重置或断电事件的情况下，该给定的存储器保护条目就留存在静态MPU存储器36中，直到另一MPU参数重新配置操作改变该条目的内容为止。这不像MPU缓存存储装置32，对于MPU缓存存储装置，即使没有用来改变给定的地址区域的实际参数组的MPU参数重新配置操作，MPU缓存存

储装置32的内容也可能根据缓存替换策略而随时间变化。例如,MPU缓存存储装置32可以与缓存控制器相关联,在需要制作用于存储器系统10中的存储器保护表22的更近访问的条目的空间时,该缓存控制器使用最近最少使用的或类似的替换策略来将被预测为将来最不有用的条目驱逐出。

[0111] 虽然图1将静态MPU寄存存储装置36示为与MPU缓存存储装置32分离,但在其他的示例中,这些也可以是共同存储结构的各个部分。例如,可以用旗标标记共同存储结构的一些条目以指示应将它们视为静态条目,不应将这些静态条目通过缓存替换策略而驱逐出。然而,实际上,提供单独的存储结构32、36可以允许更高效地对缓存存储装置32加索引。

[0112] MPU包括权限检查电路40,该权限检查电路用于读取存储在存储器系统10中(或缓存在MPU缓存存储装置32中)的存储器保护表22的查找的条目或静态MPU寄存存储装置36中指定的静态配置的MPU条目中的任一者中指定的访问权限数据,并且确定由处理电路4发出的存储器访问请求的性质是否满足由基于存储器访问请求的目标地址所选择的条目的访问权限指定的需求。例如,访问权限可指定区域是只读区域(对于只读区域,由被处理电路4执行的存储指令触发的任何写入请求都应被拒绝)还是被允许读取和写入。访问权限还可以指定是否允许取出单元5从对应的地址区域取出指令。访问权限还可以将对对应地址区域的访问限于处理电路4的某些异常等级或特权等级,并且权限检查电路40可以检查与存储器访问请求相关联的当前异常等级或特权等级是否满足访问权限中定义的限制条件。

[0113] 存储器系统10中的存储器保护表22和静态MPU寄存存储装置36可以不是影响权限检查电路40是否允许存储器访问的唯一控制信息来源。例如,装置2还可以包括安全属性单元(SAU)42,该安全属性单元包括存储器安全配置寄存器,该存储器安全配置寄存器定义地址空间中安全和非安全的区域。安全检查电路46对照寄存器44中的区域定义数据检查存储器访问请求的目标地址,并且在安全操作模式下操作时,处理电路4仅被限制访问安全区域(非安全区域在安全模式或在非安全模式中的任一者下是可访问的)。因此,存储器安全配置寄存器44可以表示用于控制对存储器的访问的控制数据结构的另一示例,该控制数据结构可以施加与MPU表或静态MPU寄存存储装置36的权限正交的权限。也就是说,会被MPU允许的访问可能还是会被SAU 42拒绝,反之亦然。可以使由处理电路的取出单元5或加载/存储单元8发起的存储器访问请求和由MPU存储器访问电路24发出以访问存储器系统10中的存储器保护表22的任何MPU存储器访问请求两者经受由SAU 42进行的安全检查。虽然在图1的示例中,哪些存储器区域被视为安全或非安全的定义仅由存储器安全配置寄存器44的内容来定义,但在其他的示例中,可以在存储器系统10内维持与MPU表22类似的存储器安全表以提供将地址区域定义为安全或非安全的另外的条目定义。这会使得能够支持大量同时定义的区域。虽然图1中将MPU 20和SAU 42示为依序操作,但一些实施方式可以针对处理电路发出的存储器访问提供对MPU和SAU并行查找,以改进性能。

[0114] 图2示出了存储在存储器系统10中的存储器保护表22的示例。存储器保护表包括许多条目50。在此示例中,每个存储器保护条目50包括存储在存储器地址空间的两个相邻数据字52中的数据。第一数据字RBARi提供对应的地址区域的基础地址(起始地址)54,而对应的地址区域的结束地址(LIMIT)56被定义在条目50的第二数据字(RLARi)中。在此示例中,访问权限(AP)58被指定在条目的RLAR数据字中,而指示条目定义了启用的地址区域还是停用的地址区域的启用旗标60是在RBAR数据字中,然而也可以将此信息58、60切换成其

他的数据字或表示在相同的数据字内。此外，条目50的每个数据字52可以可选地包括未在图2中示出的其他信息。虽然图2示出每个地址区域的起始地址54和结束地址56被表示在存储器中的单独数据字中的示例，但这并不是必需的，并且在一些实施方式中，可以将这些地址配合在单个数据字内，在这种情况下，不一定每个条目都要提供多个数据字。此外，虽然图2使用起始地址54和结束地址56来表示可变大小的地址区域，但在其他的示例中，可以由大小参数表示界限地址中的一者，该大小参数相对于另一界限地址定义对应地址区域的大小。活跃表ID寄存器28标识了存储器保护表的基础地址30，并且可选地可以标识了大小参数62，该大小参数标识了存储器保护表中的总条目数。虽然在图2中，条目50的数据字52是相邻的，但这并不是必需的，并且在一些实施方式中，可以将所有RBARi数据字一起群集在表的一部分中，而RLARi数据字被一起群集在表的另一部分中。

[0115] 存储器保护表22的条目50被分配给相同地址空间64内的地址，该地址空间用于由处理电路4的加载/存储单元8或指令取出单元5发出的常规存储器访问的地址。存储器保护条目50依据它们的起始位置54在地址空间内排序(并且也依据它们的结束地址56来排序，因为可能不允许由存储器保护表22内的存储器保护条目定义的地址区域重叠)。因此，条目0会是与具有最小起始地址的地址区域对应的条目，条目1会与具有下个最小的起始地址的地址区域对应等等。请注意，图2在地址空间64内示出的是存储表的每个存储器保护条目50的地址位置，而不是条目所针对定义访问权限的对应地址区域的位置。应理解，图2中所示的每个条目50的特定格式仅是一个示例，并且可以使用其他的格式来表示相同的信息。

[0116] 图3示出了MPU缓存存储装置32的示例，该MPU缓存存储装置包括许多缓存条目34，每个缓存条目34用于缓存来自存储器保护表22的存储器保护条目50中的一者。如图3中所示，除了对应存储器保护条目50的参数以外，缓存条目34还可以指定有效位70和表标识符72，该有效位指示缓存条目34的内容是否有效，该表标识符标识提取被缓存在该缓存条目34中的存储器保护条目50的特定的存储器保护表22(这可以与在缓存的存储器保护条目被分配给缓存32时的活跃表ID寄存器28的当前值对应)。因为很可能多个缓存条目34将具有相同的表标识符72，一些实施方式可以选择通过仅将独一的表标识符72存储在MPU缓存存储装置32内的单独表标识符表中来减少所需的存储量。在这样的实施方式中，每个缓存条目34可以在表标识符表中存储全表标识符的索引。此外，每个缓存条目34可以指定条目标识符74，该条目标识符标识了对应存储器保护表22内提取被缓存在该特定缓存条目34中的存储器保护条目50的位置。例如，在图2中所示的示例中，如果从表提取出条目ID 1并且安置在缓存中，则条目ID字段74可以指定值1。条目ID字段74可以用于处理由处理电路4执行的指令以便检查给定的地址映射到存储器保护表的哪个条目ID。通过将对条目ID的指示包括在每个缓存条目34内，这意味着，如果由指令指定的地址匹配由缓存存储装置32内缓存的存储器保护条目定义的区域中的一者，则可以简单地从缓存读取该条目的条目ID 74，而不需要从对存储器的访问确定该条目ID。

[0117] 表ID 72用来限定缓存中的查找。在查找缓存以确定缓存是否包含与给定地址对应的存储器保护条目时，只有表ID字段72匹配存储在活跃表ID寄存器28中的活跃ID的缓存条目34才可以生成命中。即使向缓存供应的查询地址是在针对对应的存储器保护条目50缓存的参数中的基础地址与界限地址之间，包含与活跃表ID不同的表ID的缓存条目34也将生成未命中。

[0118] 请注意，缓存条目34不需要具有启用旗标60的空间，该启用旗标被提供在存储在存储器保护表22内的存储器中的每个存储器保护条目50中。这是因为，缓存32可能被禁止缓存停用的存储器保护条目。在MPU缓存存储装置32中缓存的所有存储器保护条目应是启用的条目时，则启用旗标60会是缓存条目34内的冗余信息。通过禁止缓存停用的条目，这意味着，执行处理电路的软件可以包括MPU表22内停用的存储器保护条目，这些停用的存储器保护条目定义之后可以激活的区域，然后在确实需要激活这些区域使得地址访问接着将变成被允许的时候，这可以简单通过在存储器保护表22的对应存储器保护条目中设置启用旗标60来进行而不需要从MPU缓存存储装置32触发对任何缓存的无效。

[0119] 可以由执行MPU缓存无效命令的处理电路4触发来自MPU缓存存储装置32的缓存指示。MPU缓存无效命令的变型可以触发对所有缓存的条目的无效，或触发对满足由地址或地址范围、表ID、或条目ID中的任一者或更多者指定的条件的条目的无效。

[0120] 图4示出了静态MPU寄存存储装置36和MPU 20内提供的各种控制寄存器80的示例。

[0121] 对于静态MPU寄存存储装置36，图4示出了为每静态MPU条目提供的一组的三个寄存器。也就是说，对于MPU寄存存储装置内的硬件中支持的每个静态MPU条目，可以提供三个寄存器82、84、86的单独实例。寄存器86是可选的。寄存器82和84一起表示对应的静态MPU条目，该静态MPU条目具有与存储器保护表22的给定存储器保护条目50类似的内容。因此，寄存器82、84可以指定基础地址(起始地址)54和界限地址(结束地址)或大小56、访问权限58、和启用旗标60，该启用旗标控制条目是启用(能够用于检查存储器访问请求的权限)还是停用(使得不能基于此条目的内容来使存储器访问请求有效，然而在停用时，应该保留寄存器82、84、86的其余参数直到随后重新配置为止)。应理解，寄存器82、84的特定格式仅是一个示例，并且可以通过不同的字段重新布置或表示相同的信息。此外，寄存器82、84可以存储未在图4中示出的其他信息。此外，在一些情况下，可以由单个寄存器表示静态配置的MPU条目的内容。

[0122] 因此，在静态MPU条目定义与由处理电路4发送的访问请求的目标地址匹配的地址区域并且该条目启用时，则可以使用该匹配的静态MPU条目中定义的访问权限58来验证是否允许访问。在此情况下，可能不一定要访问存储器系统10以使访问请求有效。

[0123] 与由寄存器82、84表示的对应静态MPU条目相关联的额外元数据寄存器86是可选的，但如果有提供，则可以允许更高效的性能。元数据寄存器86指定寄存器停用和未知(RDU)字段88，该寄存器停用和未知字段如果被设置为1，则指示与对应静态MPU条目相关联的寄存器82、84当前不用来存储静态MPU条目。如果RDU字段88被设置为1，则这指示可以将寄存器82、84用作MPU缓存存储装置32的延伸部分来从存储在存储器系统10中的MPU表22缓存对应存储器保护条目的内容。这可以通过在软件不需要硬件中支持的全数静态MPU条目时有效地增加MPU缓存存储装置32的容量，来允许更佳的性能。元数据寄存器86的其余部分可以指定额外的缓存字段70、72、74，这些额外缓存字段被提供为用于缓存存储装置32内本来在静态寄存器中不会提供的每个缓存条目34。请注意，在使用RDU旗标88来将静态寄存器中的特定条目设置为未知时，则这与使用启用旗标60将条目设置为停用不同。在条目停用但被标记为未知时，则保留寄存器82和84的内容并且不能将这些寄存器用作缓存存储装置。例如，软件可能希望暂时停用特定的静态条目，使得不能使用该条目来使对地址空间的该区域的存储器访问有效，但保留这些寄存器82、84中的参数，使得之后可以再次启用区域

而不需要重新编程静态MPU寄存器(除了设置启用旗标60外)。在使用RDU旗标88将静态区域设置为未知时,对于被设置为未知的静态区域中缓存的条目,可以不需要启用旗标60,因为缓存可能被禁止缓存停用的存储器保护条目,如上文所讨论。因此,一些实施方式可以选择具有单个共享的有效/启用旗标而不是单独的启用旗标60和有效旗标70。

[0124] 控制寄存器80包括用于编程静态MPU条目的内容的区域编程寄存器90。控制寄存器80中的每一者是存储器映射的寄存器,可以通过执行加载/存储操作来访问该存储器映射的寄存器,该加载/存储操作将映射到该特定寄存器的存储器地址指定为目标地址。为了避免需要将地址单独分配给所提供的静态MPU条目组的每个单独条目(例如对于硬件中提供的8或16个静态MPU条目存储装置组中的每一者来说),用于静态MPU寄存存储装置中支持的条目中的每一者的相应版本的第一静态MPU寄存器82可以共享相同的存储器映射的地址(并且对于地址共享寄存器84来说是类似的)。为了选择要写入哪个特定版本的寄存器82或84,软件可以首先将区域标识符92写入到区域编程寄存器90,该区域标识符与静态MPU条目中要写入的一个特定的静态MPU条目对应,然后可以执行后续的MPU重新配置操作(将映射到寄存器82或84的地址指定为目标地址的存储指令),并且已被安置在编程寄存器90中的区域号92可以指定哪个静态MPU条目是要更新的静态MPU条目。编程寄存器90还可以包括RDU字段94,该RDU字段指定是否要通过元数据寄存器86的RDU字段88将区域标识符92在编程寄存器90中所针对指定的区域设置为停用和未知。再次地,使用在静态MPU寄存器条目组之间共享的编程寄存器90来设置RDU字段意味着,不需要针对每个元数据寄存器86提供单独的存储器映射的地址。

[0125] 如图4中所示,控制寄存器还包括活跃表ID寄存器28,在此示例中,该活跃表ID寄存器将活跃表的基础地址30指定为活跃表标识符。活跃表的大小被指定在单独的大小参数62中,该大小参数被指定在另外的控制寄存器96内。可以使用寄存器在图4中用点标记的其余字段来存储其他的信息。在此示例中,包含表的基础寄存器30的寄存器28还包括清除活跃表大小字段98。如图4的示例中,活跃MPU表22的基础地址30和大小62被存储在不同的寄存器中,这可以意味着,可能需要不同的MPU参数配置操作以分别设置基础地址和大小。因此,如果需要改变基础地址和大小,则在改变基础地址和大小中的一者与改变另一者之间可以存在一段时间,在该段时间期间,定义控制寄存器80内的活跃表的参数有效地定义可包括还未用存储器保护条目50编程的数据字的地址空间区域。这可能有对照没有旨在用来定义存储器保护条目的数据检查存储器访问操作的风险,这可能导致不确定的结果,这些不确定的结果可能中断由存储器保护表实现的安全检查。MPU缓存存储装置32的一些实施方式可以允许从存储器保护表22预先取出条目。因为这样的预先取出可以发生在任何时间点,并且可能不是由处理电路4执行的指令直接触发,可能难以确保保护表22之外的无效条目不被缓存。此问题可以通过在包含基础地址30的寄存器28中提供清除活跃表大小字段(CLRATSIZE)98来解决。在重新配置活跃表ID寄存器28时,如果要安置在该寄存器中的新数据将CLRATSIZE字段指定为等于1,则除了更新该寄存器的内容以外,还将寄存器96中的活跃表大小62清除为0,这有效地指示在存储器系统10中没有活跃表22。这防止基于下述数据字不适当地允许任何存储器访问请求:该数据字在仅已更新基础地址但还未更新大小之后被不正确地假设为存储器保护表22的一部分。后续的MPU重新配置操作可以接着将寄存器96内的地址转换大小更新为用于新的活跃表的适当值。另一方面,如果要写入寄存器28的

新值具有设置为0的CLRATSIZE字段98，则响应于此操作，只有活跃表ID寄存器28的指定基础地址30的内容会被更新，并且在此情况下，寄存器96中的大小62会保持不变。在相同大小的两个存储器保护表之间切换时，这会是更适当的。

[0126] 图5示意性地图示出由存储器保护表22的不同的基于存储器的MPU条目50或静态MPU寄存存储装置36内的静态配置的MPU条目所涵盖的地址空间的一部分。在此示例中，地址空间是物理地址空间，所以由处理电路4发出的存储器访问请求指定与存储器系统内的对应位置直接对应的物理地址，并且对照定义物理地址空间的对应区域的存储器保护条目来检查这些存储器访问请求。图5中标记的区域100与存储器保护表22的三个不同的基于存储器的存储器保护条目50对应。请注意，存储器保护表22可以定义地址空间的稀疏设置的区域，并且在由存储器保护表的连续条目定义的对应地址区域之间可以存在间隙。

[0127] 另一方面，区域102与由静态MPU寄存存储装置36表示的两个相应的静态MPU条目对应。因此，用于从存储器保护表22定位给定的基于存储器的MPU条目的访问延迟可能取决于该条目是否是在MPU缓存存储装置32中而变化，但对与静态MPU条目对应的静态区域102的访问却可以是确定的，因为静态区域将保持是被配置的，直到处理电路4执行下述指令：该指令用于重新配置定义由给定静态MPU条目定义的静态区域的访问权限或范围的参数为止。因此，软件可以确定地址空间的某些关键区域，对于这些关键区域，可能需要快速地和/或以确定性延迟处理访问，并且通过将静态MPU区域用于处理这些访问，这对于实时应用来说可以是更佳的。同时，通过支持存储在存储器中的存储器保护表22内的可变数量的存储器保护条目50，这实现了同时定义较大数量的总存储器区域，从而减少由于对于给定的地址缺乏定义的存储器保护条目所造成的错误的频率。因此，此混合方式在性能与硬件成本之间提供了更佳的平衡，特别是对于涉及实时处理或需要确定性访问延迟但具有受限的存储器容量的系统而言。

[0128] 如图5中所示，由存储器保护表22的存储器保护条目所涵盖的区域可与跟静态MPU条目对应的静态配置的地址区域重叠。在此情况下，可以用不同的方式来处理重叠。在一些实施例中，如果存储器访问请求的目标地址在静态MPU寄存存储装置36中命中，则由其匹配的静态MPU条目定义的访问权限可以优先考虑，并且否决MPU表22的匹配的基于存储器的MPU条目中指定的任何有冲突的属性。在此情况下，一旦在静态MPU寄存存储装置36中存在命中，就可以不需要由MPU存储器访问电路24发起任何存储器访问请求。

[0129] 替代地，在静态MPU寄存存储装置中存在命中并且存储器访问请求满足由该匹配的静态MPU条目指定的访问权限的情况下，则可以抑制MPU存储器访问，使得无权访问存储器保护表22，但如果存储器访问请求无法满足匹配的静态MPU条目中指定的访问权限，则仍然可以由MPU存储器访问电路24触发MPU存储器访问。如果存储器保护表22针对包括目标地址的地址区域包括匹配的MPU条目，则如果该匹配的基于存储器的MPU条目允许执行访问，则即使用于相同地址的静态MPU条目不允许存储器访问，也可以允许存储器访问。如此实现静态与基于存储器的MPU条目之间的重叠可以用于针对更高特权的进程设置能够访问更广范围的地址空间的权限，并且也可用来允许较低特权的进程访问仅访问更高特权的进程可用的地址范围的有限子集。例如，由静态MPU条目表示的区域102-1可以将访问限于较高特权的进程，使得由较低特权的代码访问的子范围100-2内的地址无法满足由静态MPU条目指定的访问权限，但可以通过基于与区域100-2相关的对应的基于存储器的MPU条目来实现的

权限检查。这避免了任何将区域102-1分成表示区域102-1的在区域100-2的任一侧的部分的单独条目的需要。

[0130] 图6是图示出检查存储器访问是否被MPU允许的方法的流程图。在步骤120处，处理电路4发出指定目标地址的存储器访问请求。存储器访问请求可以是由指令取出电路5发出的指令取出请求，或可以是由加载/存储单元8发出的数据访问请求(加载或存储操作)。在步骤122处，MPU 20确定目标地址是否命中存储在静态MPU寄存存储装置36中的静态MPU条目。也就是说，权限检查电路40检测目标地址是否在静态MPU条目中的任一者的对应地址区域内。如果是，则在步骤124处，权限检查电路确定所命中的静态MPU条目中指定的访问权限是否被满足。如果访问权限被存储器访问请求满足，则在步骤126处，存储器访问请求被MPU允许。在包括SAU 42的系统中，也可以存在由SAU 42的安全检查电路46执行的额外安全检查，以确定存储器访问请求是否被允许，因此即使MPU在步骤126处允许存储器访问请求，该请求也可能仍然被SAU 42拒绝。然而，经受SAU检查，存储器访问请求被MPU 20允许。

[0131] 如果在步骤124处，访问权限不被存储器访问请求满足，则存在MPU可以如何响应的替代选项。在第一选项中，在目标地址命中静态MPU条目但该静态MPU条目指定不被存储器访问请求满足的访问权限时，则在步骤128处，可以拒绝存储器访问请求。在此情况下，MPU可以触发对错误情况或异常进行信号告知。替代地，在第二选项中，与在步骤122处在目标地址在静态MPU寄存存储装置中未命中时相同，该方法可以继续进行到步骤130。

[0132] 因此，如果对于静态MPU寄存存储装置36中的条目中的目标地址存在未命中，或者存在命中但存储器访问请求无法满足访问权限并且系统实施选项2，则该方法继续进行到步骤130，在步骤130处，权限检查电路40检测在MPU缓存存储装置32中是否存在目标地址的任何命中。请注意，虽然一些系统可以仅在静态MPU寄存存储装置36中还不存在命中(或还不存在具有被满足的访问权限的命中)时查找MPU缓存存储装置32(即在图6中所示的顺序步骤流的情况下)，但在一些替代方案中，为了改进性能，可以与静态MPU寄存存储装置并行地查找缓存，使得万一在静态MPU寄存存储装置不能明确提供是否应允许或拒绝存储器访问请求的确定的情况下需要查找缓存，则已经发起了缓存查找，从而使得能够较早获得缓存响应。尽管如此，相对于缓存MPU条目，静态MPU条目可以被优先考虑，即使这些MPU条目是并行查找的。

[0133] 因此，如果目标地址命中(即在对应的地址区域内)缓存的MPU条目，并且该缓存的MPU条目在MPU缓存存储装置内是有效的并且与跟当前在活跃表ID寄存器28中标识为活跃表相同的MPU表对应，则在步骤132处，权限检查电路确定MPU缓存存储装置的命中条目中指定的访问权限是否被满足。如果是，则再次在步骤126处，存储器访问请求被允许且经受任何可能需要的SAU检查。如果MPU缓存存储装置32的命中条目中指定的访问权限在步骤132处未被满足，则在步骤134处，存储器访问请求被拒绝并且可以对错误进行信号告知。

[0134] 如果在步骤130处，目标地址被确定为在MPU缓存存储装置32的所有条目中都未命中(即不存在指定与活跃表ID相同的表ID并且与包括目标地址的存储器地址区域对应的缓存条目)，则在步骤136处，表查找电路26触发MPU存储器访问电路24以发起一个或更多个MPU存储器访问请求以便基于目标地址查找基于存储器的MPU表22。下文针对图7更详细地讨论了查找过程。可以针对一组预设访问权限验证和/或可以由SAU 42针对存储器安全配置寄存器44中定义的安全区域检查在查找期间由MPU存储器访问电路触发的MPU存储器访

问请求。因此，对于被发起作为MPU表查找的一部分的MPU存储器访问请求，不需要针对静态MPU寄存存储装置36或存储器保护表22检查这些存储器访问。基于表查找过程，在步骤138处，表查找电路26确定基于存储器的存储器保护表22中是否已经存在命中。如果是，则再次在步骤132处，确定存储器检测表中的命中条目中指定的访问权限是否被满足，并且根据访问权限是否被满足，接着该方法视情况继续进行到步骤126或步骤134以允许或拒绝存储器访问。除了检查存储器检测表22的命中条目中的访问权限以外，与MPU缓存存储装置32相关联的缓存控制器也可以将存储器保护表22的命中条目分配给缓存。如果需要驱逐出存储器保护表22中先前缓存的条目，则可以通过缓存替换策略选定牺牲条目，例如最近最少使用的条目。

[0135] 另一方面，如果在步骤138处，表查找电路26确定，在基于存储器的MPU表22中对于目标地址存在未命中，则在存储器保护表中并未定义与目标地址对应的地址区域，所以对未命中进行信号告知。响应于存储器保护表中的未命中，不同的选项可用于处理存储器访问请求。在一些情况下，可以简单拒绝请求并且可以对错误进行信号告知。替代地，可以定义一组预设属性，该组预设属性旨在处理不具有在存储器中的静态MPU寄存存储装置36或存储器保护表22中的任一者中定义的特定存储器保护条目的任何存储器访问请求。因此，如果在存储器保护表22中未命中，则可以对照预设属性检查存储器访问请求，然后可以由预设属性确定存储器访问请求是否被MPU 20允许或拒绝。

[0136] 图7是更详细地示出图6的步骤136处的查找过程的流程图。图7示出查找控制表的方法，该控制表包括许多条目，其中每个条目指定可变大小的对应地址区域的第一界限地址和第二界限地址与大小中的任一者。可以允许区域的大小是并非精确的2的幂的字节数。对应的地址区域的界限可以不与N字节地址边界对准，其中N是对应的地址区域的大小。在图6的步骤136中，控制表是存储器保护表22，并且用来查找表的查询地址是由处理电路4发出的存储器访问请求指定的目标地址。然而，也可以将图7的查找过程用于其他形式的控制表。例如，可以将它们使用在指定安全或非安全地址区域的控制表中、或用于追踪哪个代码属于正被执行的特定软件库的控制表中，该控制表可以由处理电路所使用以限制哪些代码区段能够调用其他软件库，这可以提供保护以免于返回定向编程攻击。因此，应理解，可以将图7中所示的方法应用于其他形式的控制表，而不是只有存储器保护表22。

[0137] 在图7的步骤150处，查找电路26接收查询地址。表查找电路26确定初始搜寻窗口，该初始搜寻窗口包括控制表22的许多条目，这些条目要被搜寻以确定查找地址是否与这些条目中的任一者的对应地址区域对应。在一些情况下，搜寻窗口可以包括控制表22的所有条目。替代地，在一些实施方式中，表查找电路26可以将不包括控制表的所有条目的更窄的搜寻窗口确定为初始搜寻窗口，其中搜寻窗口的特定窄化是基于由存储在静态MPU寄存存储单元36被标记为未知的条目中的任何缓存的条目或MPU缓存存储装置32的缓存的存储器保护条目推导的信息来确定的。例如，如果在查找MPU缓存存储装置32时，已经确定查询地址小于给定缓存条目34的基础地址，则可以知道，存储器保护表22的匹配条目可以仅具有比该缓存条目34中指定的条目ID 74小的条目标识符，因为控制表22的条目是按照基础地址来排序的。

[0138] 在步骤152处，表查找电路26开始被执行为查找表的二分搜寻过程的第一二分搜寻步骤。二分搜寻步骤(也称为搜寻窗口窄化步骤)包括如图7中所示的许多步骤152-170，

这些步骤可以在连续的二分搜寻步骤中重复多次。在步骤152处，例如在当前搜寻窗口的中间点处选择控制表22的候选条目。虽然非必需，但为了性能，可以优选地将中间点定位得尽可能靠近当前搜寻窗口的中点，然而对于搜寻窗口中的偶数量的条目，可能无法具有恰好在搜寻窗口的起始与结束之间的中间的候选条目。例如，如果搜寻窗口包括16个条目，则可以将16个条目中的条目8或条目9中的任一者选择为候选条目。此外，对于仅包括2个条目的搜寻窗口，则搜寻窗口的外边界处的条目中的一者会被选择为候选条目。

[0139] 在步骤154处，确定所选择的候选条目是否是在表的有效范围之外。因为有时候控制表22可以包括不恰好与2的幂对应的许多条目，在步骤150处，针对不具有2的幂的条目数的表，可以将搜寻窗口构造为好像条目数量等于下个最高的2的幂，并且因此有时候在步骤152处，所选择的候选条目可能实际上是在表的有效范围之外。如果是，则在步骤156处，表查找电路26抑制MPU存储器访问电路24发出用于加载所选择的候选条目的第一界限地址的任何加载操作。此处，第一界限地址可以是对应地址区域的起始地址或结束地址中的任一者，这取决于该起始地址和该结束地址中的哪个在给定实施方式中的每个二分搜寻步骤中被比较。对于上文讨论的MPU示例，使第一界限地址是对应地址区域的起始地址可能是最简单的。在步骤156处，表查找电路确定，在所选择的候选条目在表的范围之外时，可以隐含地确定查询地址与第一界限地址的比较的结果，使得在基于该比较而窄化搜寻窗口时，更窄的窗口包括表内的至少一个条目。例如，如果在步骤164处执行的比较是确定查询地址是否小于对应地址区域的基础地址，则比较的隐含结果可以是，在所选择的候选条目是在表的范围之外的情况下，查询地址小于基础地址，因为这将往往意味着，更窄的窗将排除表在有效范围之外的一部分并且将包括表中的至少一个条目。

[0140] 如果确定在步骤154所选择的候选条目是在表的有效范围内，则该方法继续进行到步骤158，在步骤158处，表查找电路26可以基于MPU缓存存储装置22或静态MPU存储器36中缓存的信息(如果标记为RDU的静态条目被保留作为缓存的话)确定是否已经知道查询地址与所选择的候选条目的第一界限地址的比较结果。例如，如果已经确定查询地址小于给定缓存条目的基础地址，则可能隐含的是，在所选择的候选条目具有大于给定的缓存的条目的条目ID 74的条目ID时，查询地址也必须小于所选择的候选条目的第一界限地址。替代地，在第一界限地址是对应地址区域的上限结束地址的系统中，则如果对缓存32的查找已经被确定的是，查询地址大于给定的缓存条目34的界限地址，则在所选择的候选条目具有等于或小于给定缓存条目34的条目ID 74的条目ID时，则可能隐含的是，查询地址必须大于所选择的候选条目的第一界限地址(结束地址)。因此，如果可以根据缓存的信息而已知通常在查询地址与所选择的候选条目的第一界限地址之间执行的比较的结果，则在步骤160处，用于加载所选择的候选条目的第一界限地址的加载MPU存储器访问请求被表查找电路26抑制，相反可以使用由缓存的信息确定的比较结果，而不是实际执行针对从存储器系统10加载的第一界限地址比较查询地址的操作。请注意，步骤158是可选的，并且因此一些实施方式可以不使用缓存的信息来引导查找过程。此外，在系统中，如果在步骤150处基于存储在MPU 20内的缓存的条目来窄化搜寻窗口，则在该阶段处窄化的窗口可以已经排除由缓存存储装置所知道的不需要的条目，并且因此在此情况下，步骤158可以不是必要的，因为这可能已经有效地在步骤150处考虑了。因此，应理解，可以择一在制定初始搜寻窗口时和/或在检查给定二分搜寻步骤的比较结果时，在两个步骤中的一者处考虑缓存的信息，但进

行上述步骤中的任一者并不是必需的。为了完整起见，在图7中示出了两者。

[0141] 如果不能基于已经缓存的信息来确定查询地址与所选择的候选条目的第一界限地址之间的比较结果，则在步骤162处，表查找电路26触发MPU存储器访问电路24来触发加载MPU存储器访问请求以便从存储器系统加载所选择的候选条目的第一界限地址。因此，可以发出加载请求，该加载请求将与活跃MPU表的基础地址30与基于所选择的候选条目的条目标识符来确定的偏位的总和对应的地址指定为该加载请求的目标地址。可以使此加载操作经受基于预设的属性组和/或由SAU 42指定的存储器安全配置进行的权限检查。在针对所选择的候选条目返回了第一界限地址(对应的地址区域的起始或结束地址)时，则在步骤164处，将查询地址与返回的第一界限地址进行比较。无论是否实际上在步骤164处执行比较或是否在步骤160处基于缓存的信息隐含地确定比较结果，在步骤166处，窄化搜寻窗口以排除不满足比较结果的条目。例如，在第一界限地址是对应的地址区域的起始(基础)地址的情况下，步骤164处的比较可以确定查询地址是否小于第一界限地址。如果查询地址小于第一界限地址，则在步骤166处，可以确定更窄的搜寻窗口为包括先前搜寻窗口的包括具有比所选择的候选条目的条目标识符小的条目标识符的条目的部分。可选地，也可以将所选择的候选条目本身包括在更窄的搜寻窗口中，这有时候可以通过减少关键时序路径的长度来使得电路实现更快速，因为这避免了用来将所选择的候选条目的条目标识符减一以标识更窄搜寻窗口的顶部的额外减法操作的需要。如果查询地址大于所选择的候选条目的起始地址，则更窄的搜寻窗口可以排除具有比所选择的候选条目的条目标识符小的条目标识符的条目。

[0142] 另一方面，如果第一界限地址是对应地址区域的结束地址，并且在步骤164处，比较是确定查询地址是否大于第一界限地址，则如果查询地址大于第一界限地址，则在步骤166处，更窄的搜寻窗口可以包括具有大于所选择的候选条目的条目标识符的条目标识符的这些条目。另一方面，在查询地址小于对应地址区域的上限地址时，则可以窄化搜寻窗口以包括所选择的候选条目和具有比所选择的候选条目的条目标识符小的条目标识符的任何其余条目。

[0143] 在步骤168处，表查找电路确定择一在步骤156处或在步骤166处执行的窄化之后，更窄的搜寻窗口现在是否仅包括控制表的单个候选条目。如果否，则仍然需要进一步的二分搜寻步骤，并且在步骤170处，该方法继续进行到下个二分搜寻步骤，该下个二分搜寻步骤将由步骤166或步骤158所带来的窄化的搜寻窗口用作当前搜寻窗口。该方法接着移回步骤152以重复步骤152到168以便进行另一二分搜寻步骤。

[0144] 如果在步骤168处，确定更窄的搜寻窗口确实包括单个条目，则不需要进一步的二分搜寻步骤。在此情况下，在步骤172处，从存储器系统加载由更窄的搜寻窗口中剩余的单个条目指定的第二界限地址或大小。再次地，可以使被发出以从存储器取出单个条目的第二界限地址或大小的加载MPU存储器访问请求经受由SAU 42进行的检查和/或基于预设的存储器访问属性进行的检查。在一些情况下，MPU存储器访问电路24也可能需要触发加载，以从存储器系统获得单个候选条目的第一界限地址，即使在已经在先前的二分搜寻步骤处检查该第一界限地址时也是如此。其他的实施方式可以提供在较早的二分搜寻步骤中读取的一个或更多个选定的候选条目的第一界限地址的缓存，并且因此如果在步骤172处剩余的单个条目的第一界限地址仍被缓存，则可以抑制用来获得单个条目的第一界限地址的加

载。

[0145] 在步骤174处，查找电路26确定查询地址是否是在针对单个剩余的候选条目的对应地址区域定义的界限内。因此，如果通过地址指定第二界限，则查找电路确定查询地址是否位在第一界限地址与第二界限地址之间(第一界限地址和第二界限地址是否被视为是对应地址区域的一部分或是否这些地址中的一者可能被视为在对应地址区域之外可以是取决于实施方式的选择)。另一方面，如果使用大小来定义第二界限，则查找电路可以将第一界限地址加上或减去该大小，以确定第二界限地址，然后再次确定查询地址是否位在第一界限与第二界限之间。如果查询地址是在针对与单个剩余的条目对应的区域定义的界限内，则在步骤176处，在存储器保护表中检测到命中。可以接着返回和使用单个剩余的条目中指定的属性以控制由处理组件执行的给定操作。例如，对于上文示出的MPU示例，使用命中条目来执行的操作可以是在图6的步骤132处检查访问权限，但对于其他类型的控制表，可以使用返回属性来执行不同的操作。另一方面，如果查询地址不在单个剩余的条目的对应地址区域的界限内，则在步骤178处检测到未命中，因为不存在与包括查询地址的地址区域对应的匹配条目。

[0146] 因此，在利用图7中定义的二分搜寻过程的情况下，仅需要针对在步骤168之后剩余的单个条目从存储器加载定义第二界限地址或大小的参数。对于每个二分搜寻步骤，仅需要在步骤162处从存储器加载第一界限地址(并且有时候，如果可能已经使用缓存的信息来确定将查询地址与所选择的候选条目的第一界限地址进行比较的结果会是什么，则无论如何都可以抑制此加载)。此方法有助于减少查找控制表所需的对存储器的访问的次数，从而允许较快速的性能，因为最大存储器开销可以与条目N的数量与log2(N)+1成比例，而不是与log2(N)*2成比例。

[0147] 图8示出了表具有16个条目的示例的二分搜寻过程的具体示例。被标记为200的每个步骤表示图7的特定二分搜寻步骤152-170，并且标示有特定的比较，该特定的比较会在对应的二分搜寻步骤的步骤164处执行。在此示例中，第一界限地址是对应地址区域的基础地址，并且因此在每个二分搜寻步骤处，在该步骤处在查询地址与所选择的候选条目的基础地址之间作出小于比较。例如，注记B6指的是候选条目6的基础地址。因此，在第一二分搜寻步骤200-0中，初始搜寻窗口(SW)可以包括所有16个条目0-15，并且所选择的候选条目例如是条目8。基于查询地址与所选择的候选条目8的基础地址B8之间的小于比较，接着如果查询地址小于条目8的基础地址，则将搜寻窗口窄化到条目0-7，或如果查询地址等于或大于条目8的基础地址，则将搜寻窗口窄化到条目8-15。类似地，后续的二分搜寻步骤再次作出候选条目的二分分离，以便在每个二分搜寻步骤处将搜寻窗口窄化一半，并且因此在最多四次的二分搜寻步骤中，可以将16个条目的搜寻窗口减少到如图8中所标记的步骤202处所标识的单个条目。在二分搜寻步骤200中的一些处的比较的比较结果可以被隐含地确定，而不是实际通过加载第一界限地址并且将该第一界限地址与查询地址进行比较来确定，条件是可以基于缓存的信息知道此比较结果，或条件是表中的实际条目数不是2的幂且因此上部条目中的一些可能实际上不具有所提供的有效条目。例如，如果表仅包括13个条目，则标记为200-1并且用于将查询地址与条目14的基础地址进行比较的二分搜寻步骤可以隐含地提供“是”的结果，以便接着直接继续进行到对包括条目12和13的更窄搜寻窗口作用的二分搜寻步骤。类似地，对于N＝13(即存在13个条目0到12)，因为顶部条目是条目12而没有条

目13，接着标记为200-2的二分搜寻步骤也可以产生隐含的“是”的结果，以标识剩余在搜寻窗口中的单个条目应是条目12。

[0148] 因此，在已经在步骤202中的一者中标识单个剩余的候选条目的情况下，这意味着不是查询地址命中单个剩余的条目，就是查询地址完全未命中。为了确定这些情况中哪个是真的，在步骤204(即图7的步骤174)处，查找电路确定查询地址是否在由单个剩余的条目定义的区域的基础地址与界限地址之间。应理解，可以通过大小参数来确定基础与界限中的一者或另一者，该大小参数应用于区域的另一界限。因此，如果查询地址是在对应条目的基础地址与结束地址之间，则在步骤206处，在该特定条目中检测到命中，并且如果查询地址不在这些界限之间，则在步骤208处检测到未命中。

[0149] 图9示出了在表中查找特定查询地址的具体示例。在此示例中，初始搜寻窗口被确定为包括七个条目(条目0-6)，因为使用现有的缓存的信息或静态MPU寄存器信息，或者因为表首先仅包括七个条目。在第一二分搜寻步骤200-8中在搜寻窗口的中间点处选定条目3，并且这确定的是，查询地址0x00880小于条目3的起始地址0x07500，并且因此搜寻窗口窄化到排除条目4-6(并且可选地，取决于所使用的特定实施方式，也可以根据需要排除或包括条目3)。在第二二分步骤200-10中，所选择的候选条目是条目1，并且此时，查询地址被确定为大于条目1之后的区域的起始地址。因此，现在搜寻窗口可以排除条目0。因此，现在知道的是，查询地址必须与条目1或条目2对应，或者不与表中的条目中的任一者对应。因此，二分搜寻继续，并且现在条目2在步骤200-12处变成所选择的候选条目并且加载区域2的基础地址。此时，查询地址与区域2的基础地址的比较指示的是，查询地址小于区域2的基础地址。因此，确定的是，区域1是正确的加载区域，并且返回此条目，并且从存储器加载定义该区域的结束地址的界限地址。如果处理器已经在先前的二分搜寻步骤处缓存了区域1的基础地址的值，则可以避免重新加载区域1的基础地址的需要。在此特定的情况下，查询地址接着被证明是在区域1的起始地址与结束地址之间，并且因此在存储器检测表中检测到命中。然而，如果查询地址还不是在最终二分搜寻步骤中标识的最终条目的界限之间，则会检测到未命中。

[0150] 图10示出了惰态保存的示例，该惰态保存可以用于处理图1中所示的处理系统2内的异常。如图1中所示，处理电路可以包括通用寄存器12和浮点寄存器14。在遭遇异常时，可能需要将寄存器12的内容保存到提供在存储器10中的堆栈数据结构，使得可以保留在异常之前执行的软件的架构状态，使得在异常之后执行的软件(例如异常处理器或已经被进行上下文切换的进程)可以将数据重写于寄存器12并且接着重新使用这些寄存器。然而，虽然也可能需要保留寄存器14中的浮点架构状态，但实际上，并非所有的软件进程都使用浮点寄存器。在不包括任何浮点指令的进程中，则可能不需要浮点寄存器。将来自寄存器12、14的状态保存到存储器系统10中的堆栈数据结构需要一些时间。因此，如果在异常之前执行的第一软件进程或在异常之后执行的第二软件进程中的任一者不需要浮点寄存器，则可能实际上浪费了花在从存储器保存浮点状态的时间。

[0151] 因此，如图10中所示，可以使用惰态保存方法，其中响应于异常，将整数寄存器12保存到存储器中的堆栈，但不响应于异常而保存寄存器14中的浮点状态。而是，在已经确定第一软件已经使用了浮点寄存器的情况下，仅在第二软件随后执行浮点指令以触发对浮点寄存器14的访问时，才可以触发浮点寄存器的保存。虽然不响应于异常而实际保存浮点寄

存器，尽管如此，响应于异常，在存储器10中的堆栈上为浮点寄存器保留了一些空间，并且记录堆栈上的保存区域的地址。这确保，无论是否已经保存了浮点寄存器，用以保留针对在异常之前执行的第一软件进程所保存的状态的堆栈框架的布局可以是一致的，这可以使得后续的状态恢复更简单。

[0152] 然而，在保留地址来保存浮点寄存器时，此地址可以是堆栈数据结构中可被在异常发生之前执行的第一软件进程访问的地址。在将浮点寄存器实际上保存到堆栈中的保留空间之前，MPU 20的权限检查电路40可能需要检查用来将浮点状态保存到堆栈中的保留空间的存储器访问是否被允许，然而因为可能已经将活跃表标识符寄存器28重新配置为用于由第二软件进程使用，MPU 20可能不再有权访问由第一软件使用的MPU表，该MPU表定义是否允许第一软件将状态保存到堆栈上的该区域。

[0153] 为了解决此问题，在发生异常时，除了记录在堆栈上为了浮点寄存器保留的空间的地址以外，与处理电路4相关联的异常处理逻辑也可以记录MPU检查下述信息：该信息使得硬件在第二软件进程的执行期间能够验证堆栈上的保留空间的地址是否满足在第一软件的执行期间使用的MPU表22的对应条目中定义的访问权限。此记录的MPU检查信息可以采取不同的形式。在一些情况下，所记录的MPU检查信息可以包括在异常发生时存在于寄存器28中的活跃表标识符，使得MPU可以接着之后在保存浮点寄存器时，检查地址是否满足该特定存储器保护表中指定的访问权限，即使在活跃表已经由于异常而改变时也是如此。替代地，MPU可以在异常本身的时候预先验证为了存储浮点状态而保留的区域，并且记录指示该验证是否成功的信息。图11A和图11B示出了使用第一选项的示例，而图12A和12B示出了使用这些选项中的第二者的示例。

[0154] 应理解，图10中所示的惰态仅是用于处理异常的一个选项。在一些系统中，控制寄存器内的控制数据可以指定是否实际上要将惰态保存用于处理异常，所以可能不总是将惰态保存方式应用于每个异常，并且可能仍然通过在异常时保存整数和浮点状态两者来处理一些异常。因此，可以在惰态保存被控制数据允许的情况下使用图11a到图12b中所示的技术。

[0155] 图11A和图11B示出了在遭遇到要在异常之后执行的浮点指令时执行用于浮点状态的堆栈上的保留空间的地址的验证的情况下的惰态保存的示例。图11A示出了响应于异常本身而触发的动作，并且图11B示出了由后续的浮点指令触发的动作。

[0156] 如图11A中所示，在步骤250处，响应于异常，异常处理硬件触发将整数寄存器保存到堆栈数据结构。在步骤252处，将空间分配于堆栈上以用于保存浮点状态(例如更新堆栈指针数据，使得任何随后保存的数据都将被保存到在针对浮点状态所分配的空间之外的堆栈位置，即使在还未保存浮点状态本身时也是如此)。替代地，一些实施例可以反向执行步骤252和250，使得将整数寄存器保存到针对浮点状态分配的空间之外的堆栈。通过在控制寄存器中设置旗标来将惰性浮点保存操作标记为是待处理的(pending)。此外，将标识浮点保存区域的地址保存到存储器或控制寄存器。在步骤254处，异常处理硬件触发再次将MPU检查信息保存到存储器中的结构或寄存器中的任一者，该MPU检查信息用于使得硬件之后能够检查保留的浮点保存区域的地址是否可以有效地被在异常之前执行的进程访问。在此示例中，MPU检查信息可以例如是在异常发生时在寄存器28中提供的活跃表ID，或可以是某个其他的标识符，该标识符使得能够确定与在异常之前执行的进程相关联的MPU表中的所

需条目的位置。应理解，图11A并未示出可能响应于异常而发生的所有操作，而是仅示出与惰态保存相关的操作。在一些情况下，可以与堆栈相关联的MPU条目可锁定在缓存中以防止驱逐出。

[0157] 如图11B中所示，在已经处理了异常并且在异常之后开始执行第二软件进程的情况下，接着遭遇到浮点指令。在步骤260处，确定存储在控制寄存器中的旗标是否指示浮点保存操作是否仍然是待处理的。如果浮点保存操作不再是待处理的，则在步骤262处，执行浮点指令。然而，如果浮点保存操作仍然是待处理的，则在步骤264处，获得保留的浮点保存区域的地址，并且通过MPU使用所保存的MPU检查信息来检查该地址。可能需要检测是否可以基于通过MPU检查信息标识的MPU表验证为了保存浮点保存而保留的整个地址区域。如果为了保存浮点状态而保留的区域的起始地址和结束地址两者与MPU表的相同存储器保护条目对应，则这可以简化检查，因为针对该区域内的一个地址进行的访问权限的单次检查可能足以确定是否允许用于将浮点状态保存到堆栈的所有访问。因此，在一些情况下，在图11A的步骤254处所存储的MPU检查信息也可以包括对保留区域的起始地址和结束地址是否与相同的存储器保护条目对应的指示。

[0158] 因此，在步骤264处，MPU 20获得与保留区域对应的一个或更多个存储器保护条目，并且在步骤266处检查由这些条目指定的访问权限是否被为了保存浮点状态发出的存储器访问请求满足。如果访问权限被满足，则在步骤268处，将浮点状态保存到存储器中的堆栈结构，并且清除浮点保存待处理指令以指示不需要对后续的浮点指令再次执行状态保存。然而，如果对于用于保存浮点状态的任何访问请求而言，权限未被满足，则在步骤270处，对错误进行信号告知。在一些情况下，错误可以防止执行浮点指令。然而，在其他情况下，尽管有错误，也可以执行浮点指令，因为如果错误具有比已经在处理的异常低的优先度，则错误可能仍然是待处理的，直到已经完成处理该先前的异常为止，并且可以接着处理该错误。图11B的步骤270与262之间的虚线指示，有时候，尽管有错误，仍然可以执行浮点指令。

[0159] 图12A和12B图示出图11A和图11B的替代方案，其中图12A示出了响应于异常的状态保存，并且图12B示出响应于遭遇到的浮点指令而针对状态保存执行的操作。在此示例中，响应于异常，步骤250与图11A中的相同。在步骤282处，在堆栈结构中分配空间以用于保存浮点寄存器。再次地，一些实施方式可以保留这些步骤并且在为了保存整数寄存器分配空间之前在堆栈上针对浮点寄存器分配空间。在此示例中，在图12A的步骤284处的异常处理时预先验证堆栈上的分配区域的地址。也就是说，针对由活跃表ID寄存器28标识的活跃存储器保护表验证堆栈中的分配空间内的位置的地址。基于任何匹配的MPU条目中指定的访问权限，确定用于保存浮点状态的访问是否会被允许，并且如果是，则在步骤286处，将浮点保存区域的地址存储到存储器或寄存器。在一些情况下，已经存储浮点保存区域本身的地址的事实可能足以指示，步骤284处的预先验证是成功的，所以可能不需要也存储指示预先验证是成功的旗标。替代地，也可以提供单独的旗标以确认验证是成功的。在已经在步骤286处保存为了保存浮点状态分配的区域的地址的情况下，在步骤288处，设置浮点保存待处理指令以指示，会需要对在异常之后执行的第一浮点指令进行浮点保存操作。

[0160] 因此，在此示例中，MPU检查信息可以是步骤284处的预先验证是成功的指示。

[0161] 如果步骤284处的预先验证确定，不允许用于保存浮点状态的访问，则可以有2个

选项。在第一选项中，在步骤290处，异常处理电路或MPU可以设置指定用于存储浮点状态的访问会是无效的控制旗标，但尽管如此，未触发错误，然后在步骤288处，浮点保存操作被标记为是待处理的。可以优选地允许即使预先验证不成功，也在没有错误的情况下继续进行后续的处理，因为这意味着，如果在异常之后执行的代码从不访问浮点寄存器，则因为不会不必要地触发错误而改进了性能，因为从不执行对用于存储浮点状态的保存区域的无效访问。替代地，在第二选项中，在步骤292处，在预先验证中确定不允许访问时的情况下，可以触发错误。

[0162] 图12B示出了响应于在处理异常之后遭遇浮点指令而执行的对应操作。在步骤260处，确定浮点状态保存操作是否是待处理的，并且如图11B，如果不存在待处理的浮点保存操作，则在步骤262处，如前述地执行浮点指令。如果浮点保存操作是待处理的，则在步骤294处确定是否已经保存了浮点保存区域的地址和是否已经在预先验证中将访问标示为无效的。在一些情况下，可以将地址本身已经被存储的事实采用作为访问还未被标记为是无效的指示，或者如果特定的控制旗标指定步骤284处的预先验证是否成功，则这可以用来确定是否继续进行浮点保存。如果已经保存浮点保存区域的地址并且访问不是无效的，则在步骤296处，将浮点状态保存到如由浮点状态保存区域的地址标识的先前保留在堆栈上的区域。此外，将浮点保存操作清除为是待处理的，使得任何后续的浮点指令将不会触发浮点保存操作。另一方面，如果访问被标记为是无效的或还未保存浮点保存区域的地址，则在步骤298处，触发错误。请注意，这与图12A的示例中的选项1对应，因为如果已经选择了选项2，则处理永远不会到达图12B。

[0163] 因此，通常而言，通过保存一些信息来实现不是定位与在异常之前执行的进程相关联的相关MPU条目，就是确认在响应于异常时执行的预先验证是成功的，这使得能够安全地验证在异常之后执行进程的同时由硬件触发的惰态保存，即使在活跃存储器保护响应于异常而改变时也是如此。

[0164] 图13图示出可使用的模拟器实施方式。虽然先前所述的实施例是从用于操作支持相关技术的特定处理硬件的装置和方法的方面实施本发明，但也可能提供根据本文中所述的实施例的指令执行环境，该指令执行环境是通过使用计算机程序来实现的。这样的计算机程序通常称为模拟器，只要它们提供硬件架构的基于软件的实施方式。模拟器计算机程序的变型包括模拟器、虚拟机、模型、和二进制转换器(包括动态二进制转换器)。通常而言，模拟器实施方式可以运行在支持模拟器程序310的主机处理器330(可选地运行主机操作系统320)上。主机处理器330包括主机处理电路332、存储器管理单元(MMU)334、和存储器系统336，该存储器管理单元基于存储在存储器中的层级页表支持地址转换和存储器访问权限检查。在一些布置中，可能在硬件与所提供的指令执行环境之间存在着多层模拟，和/或存在着提供在相同主机处理器上的多个相异的指令执行环境。历史上，已经需要强力的处理器来提供用合理的速度执行的模拟器实施方式，但这样的方法在某些情况下可以是合理的，例如在基于兼容性或重复使用的理由而需要运行相对于另一个处理器而言是原生的代码时。例如，模拟器实施方式可以提供具有未受主机处理器硬件所支持的额外功能的指令执行环境，或提供通常与不同的硬件架构相关联的指令执行环境。Robert Bedichek在1990年冬季USENIX会议第53-63页的“Some Efficient Architecture Simulation Techniques”中给出了模拟的概述。

[0165] 在先前已经针对特定硬件构造或特征描述实施例的情况下，在模拟的实施例中，可以由合适的软件构造或特征提供等效的功能。例如，可以在模拟的实施例中将特定的电路实现为计算机程序逻辑。类似地，可以在模拟的实施例中将存储器硬件(例如寄存器或缓存)实现为软件数据结构。在先前所述的实施例中所指称的硬件元件中的一者或更多者存在于主机硬件(例如主机处理器330)上的布置中，一些模拟的实施例可以在合适时利用主机硬件。

[0166] 模拟器程序310可以被存储在计算机可读存储介质(其可为非暂态介质)上，并且提供与目标代码300(其可包括应用程序、操作系统、和超管理器)的程序接口(指令执行环境)，该程序接口与正被模拟器程序310所建模的硬件架构的应用程序编程接口相同。因此，可以使用模拟器程序310来通过在指令执行环境内执行目标代码300的程序指令，使得实际上不具有上文讨论的装置2的硬件特征的主计算机330可以模仿这些特征。

[0167] 如图13中所示，模拟器程序310可以包括在功能上与图1的处理电路4、MPU 20、和SAU 42对应的处理程序逻辑340、MPU程序逻辑350、和SAU程序逻辑360。MPU程序逻辑350包括在功能上分别与权限检查电路40、MPU存储器访问电路24、和表查找电路26对应的权限检查程序逻辑352、MPU存储器访问程序逻辑354、和表查找程序逻辑356。MPU程序逻辑350也包括用于管理分别与静态MPU寄存存储装置36和活跃表ID寄存器28对应的静态MPU条目存储结构357和活跃ID数据结构358的程序逻辑。数据结构357、358的实际数据被存储在主机硬件330的存储器系统336中，但模拟器程序包括用于定义数据结构357、358和访问数据结构357、358上的数据的逻辑。不需要提供与图1的MPU缓存存储装置32对应的程序逻辑，因为在模拟时，所有下层数据都被存储在主机硬件330的存储器系统336中，并且因此模拟缓存不大可能提供任何性能改进。尽管如此，与MPU表22分离地模拟静态MPU条目数据结构357，使得模拟器保持与被设计为与静态定义的MPU条目交互作用的代码相容。在一些实施方式中，模拟器310也可以包括用来模拟MPU缓存存储装置32的程序逻辑，因为虽然这可能无益于模拟的性能，但如果模拟器旨在准确地模拟被模拟的MPU 350会用来响应存储器访问请求的频率循环数，则可能需要这样。SAU程序逻辑360包括在功能上与图1的存储器安全配置寄存器44和安全检查电路46对应的存储器安全配置数据结构362和安全检查程序逻辑364。

[0168] 模拟器程序310管理所模拟的地址空间370的模拟，该地址空间与会在由模拟器程序310所模拟的处理器上提供的物理地址空间对应。要执行的目标代码300、和存储器保护表22与所模拟的地址空间370内的地址相关联。目标代码300包括指定所模拟的地址空间370内的物理地址的存储器访问指令，这些指令经受由MPU程序逻辑350基于静态MPU条目数据结构357和/或MPU表22中定义的访问权限进行的权限检查，并且如果它们通过权限检查则在模拟器程序310的控制之下被映射到主机平台的虚拟地址，这些虚拟地址可以接着被主机硬件的MMU 334转换成用来访问主机存储器系统336的物理地址(MMU 334可以应用进一步的权限检查基于由主机操作系统320定义的页表来检查对存储器系统的存储器访问是否被允许)。

[0169] 因此，模拟器310允许硬件上不支持具有基于存储器的存储器保护表的MPU的主机平台执行针对具有这样的硬件支持的平台所设计的代码。

[0170] 在本申请中，用词“被配置为...”用来意指装置的元件具有能够实现所定义的操作的配置。在此上下文下，“配置”意指硬件或软件互连的布置或方式。例如，装置可以具有

提供所定义的操作的专用硬件，或可以将处理器或其他处理设备编程为执行该功能。“配置为”并不暗示，需要用任何方式改变装置元件以提供所定义的操作。

[0171] 虽然已经参照附图而在本文中详细描述了本发明的说明性实施例，但应理解，本发明不限于这些精确的实施例，并且本领域技术人员可以在这些实施例中作出各种改变和修改而不脱离如由所附权利要求定义的本发明的范围和精神。

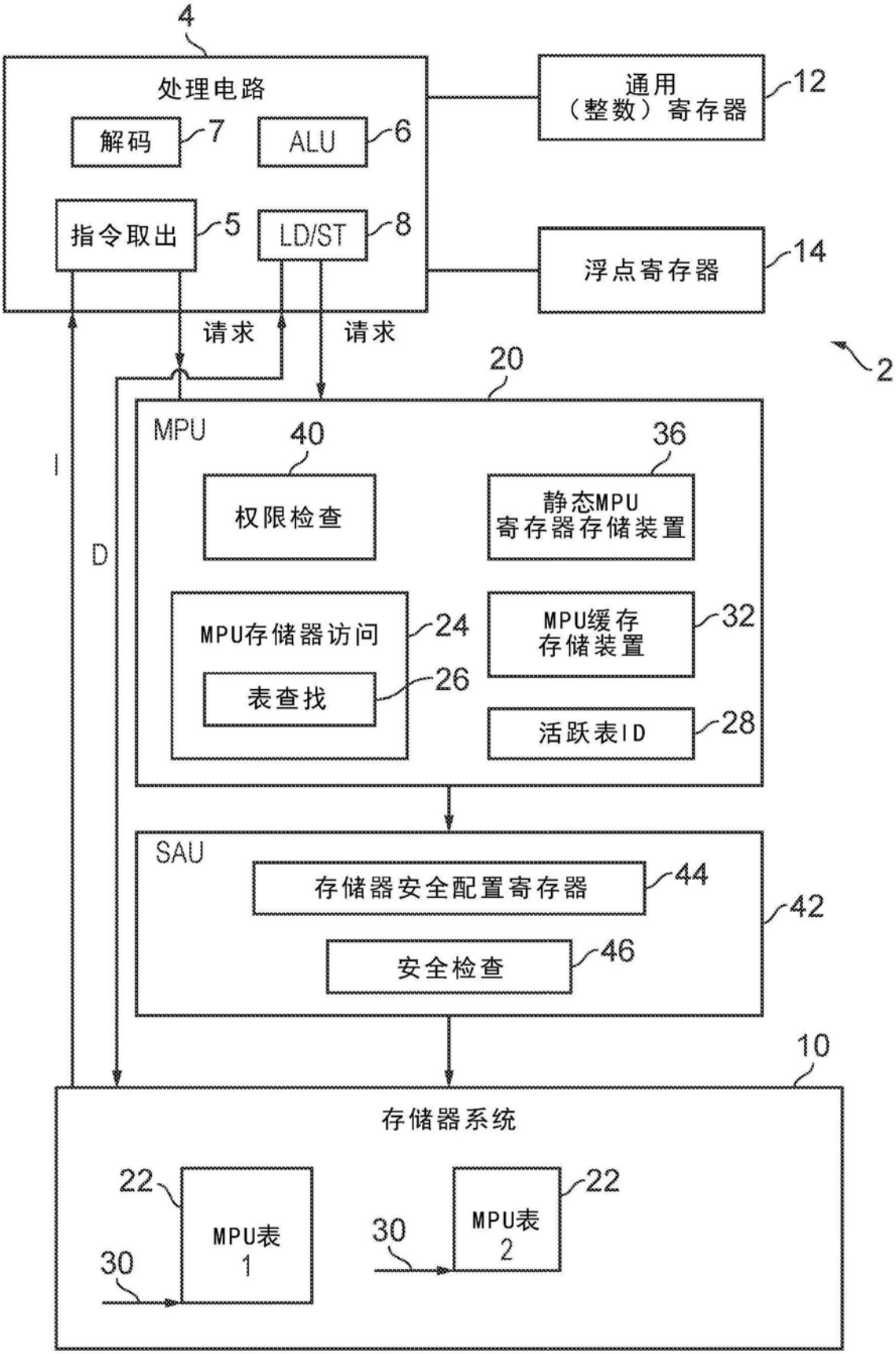

4
处理电路
解码
7
ALU
6
指令取出
5
LD/ST
8
通用
（整数）寄存器
12
浮点寄存器
14
请求
请求
2
20
MPU
40
权限检查
36
静态MPU
寄存器存储装置
D
MPU存储器访问
24
表查找
26
MPU缓存
存储装置
32
活跃表ID
28
SAU
存储器安全配置寄存器
44
42
安全检查
46
10
存储器系统
22
MPU表
1
30
30
MPU表
2
22

图1

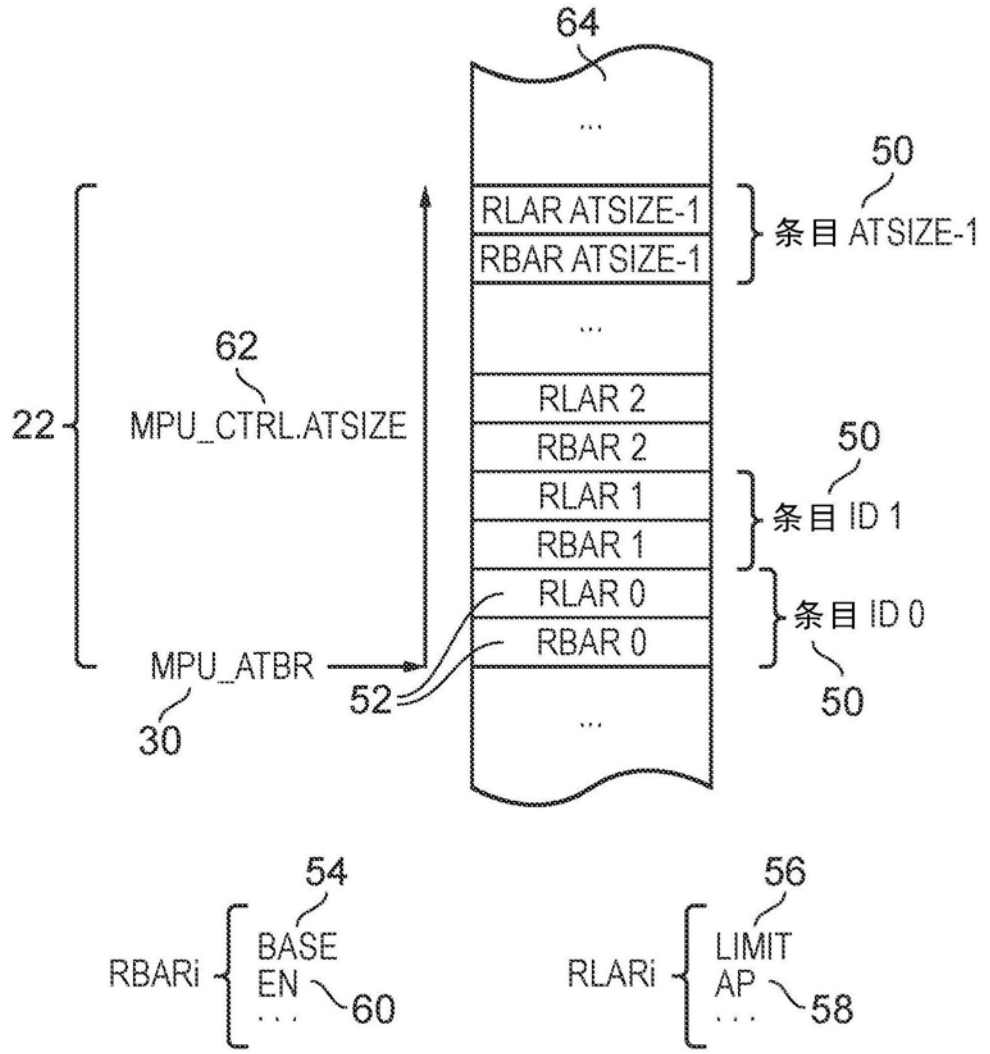

64
...
50
RLAR ATSIZE-1
RBAR ATSIZE-1
条目 ATSIZE-1
...
62
22
MPU_CTRL.ATSIZE
RLAR 2
RBAR 2
50
RLAR 1
RBAR 1
条目 ID 1
RLAR 0
RBAR 0
条目 ID 0
MPU_ATBR
52
50
30
...
54
56
RBARi
BASE
EN
...
60
RLARi
LIMIT
AP
...
58

图2

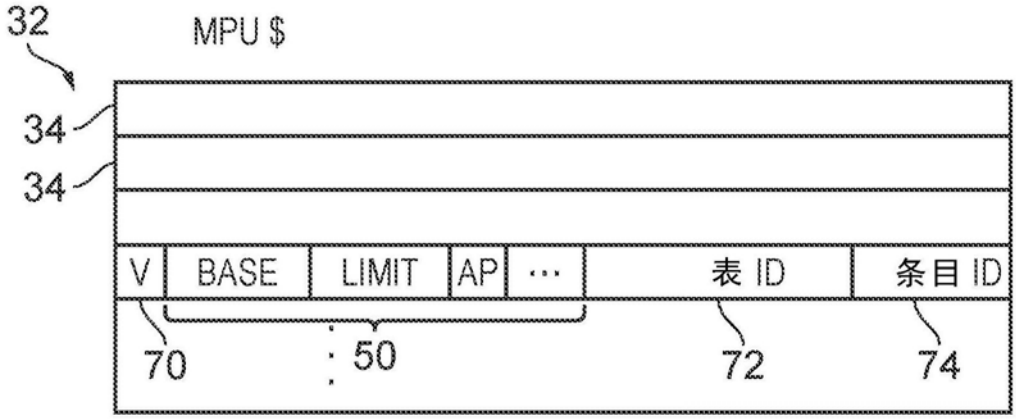

32
MPU $
34
34
V
BASE
LIMIT
AP
...
表 ID
条目 ID
70
50
72
74

图3

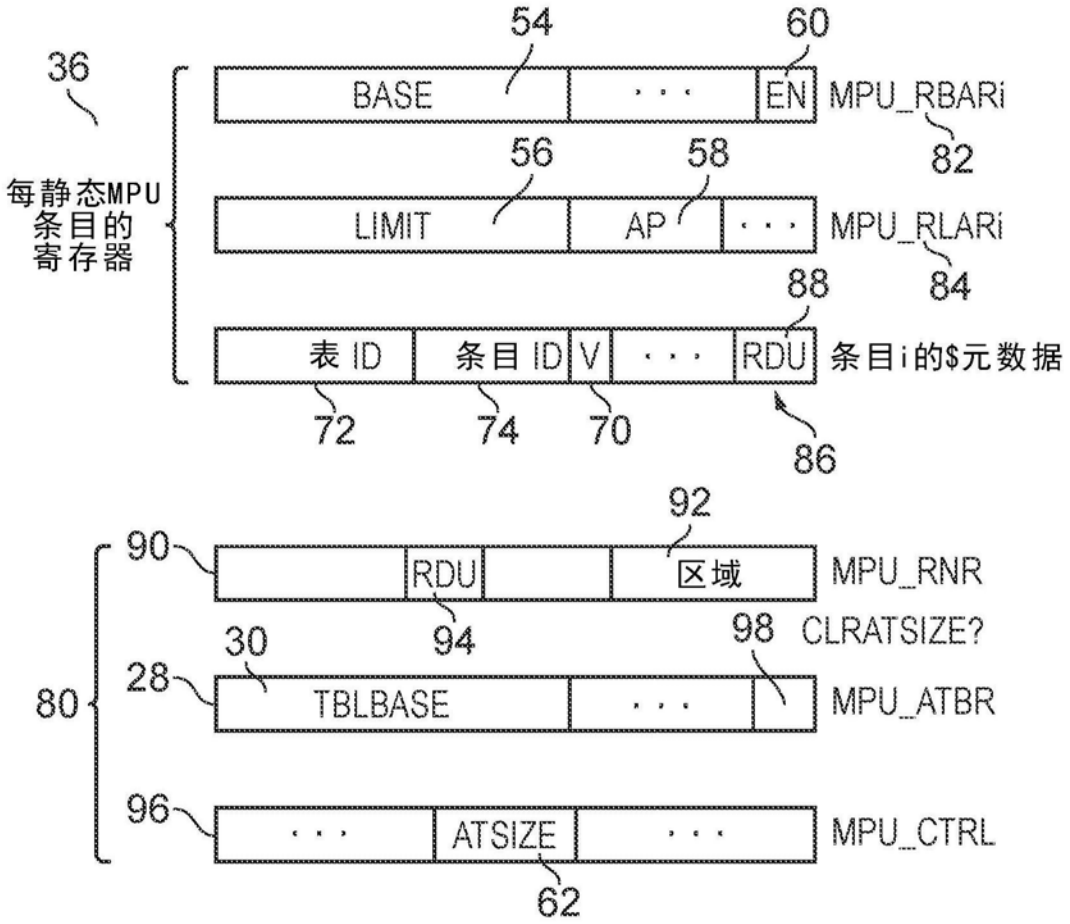

36
每静态MPU条目的寄存器
54
60
BASE
EN
MPU_RBARi
82
56
58
LIMIT
AP
MPU_RLARi
84
88
表 ID
条目 ID
V
RDU
条目i的$元数据
72
74
70
86
92
90
RDU
区域
MPU_RNR
94
30
98
CLRATSIZE?
28
80
TBLBASE
MPU_ATBR
96
ATSIZE
MPU_CTRL
62

图4

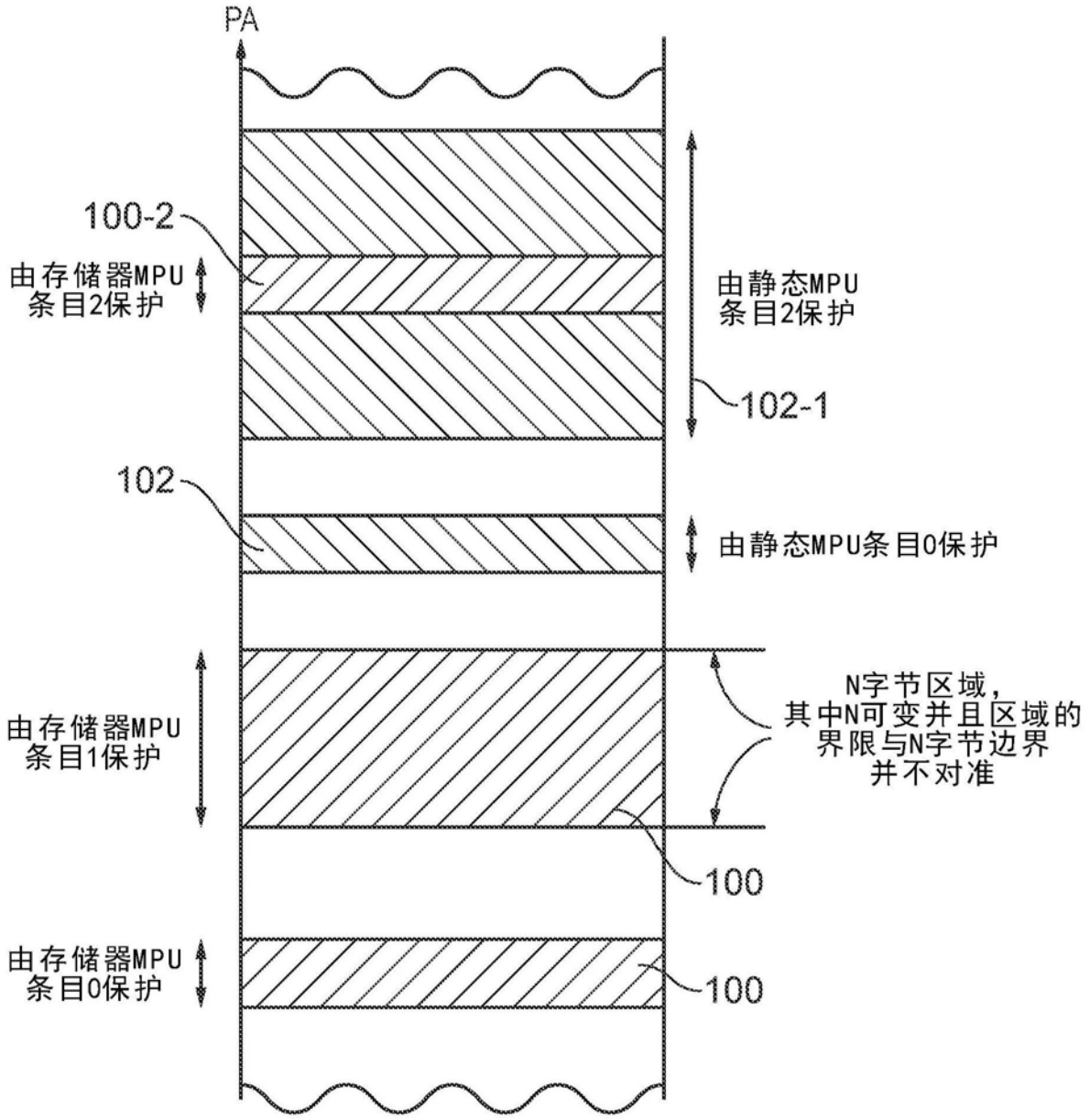

PA
100-2
由存储器MPU
条目2保护
由静态MPU
条目2保护
102-1
102
由静态MPU条目0保护
由存储器MPU
条目1保护
N字节区域，
其中N可变并且区域的
界限与N字节边界
并不对准
100
由存储器MPU
条目0保护
100

图5

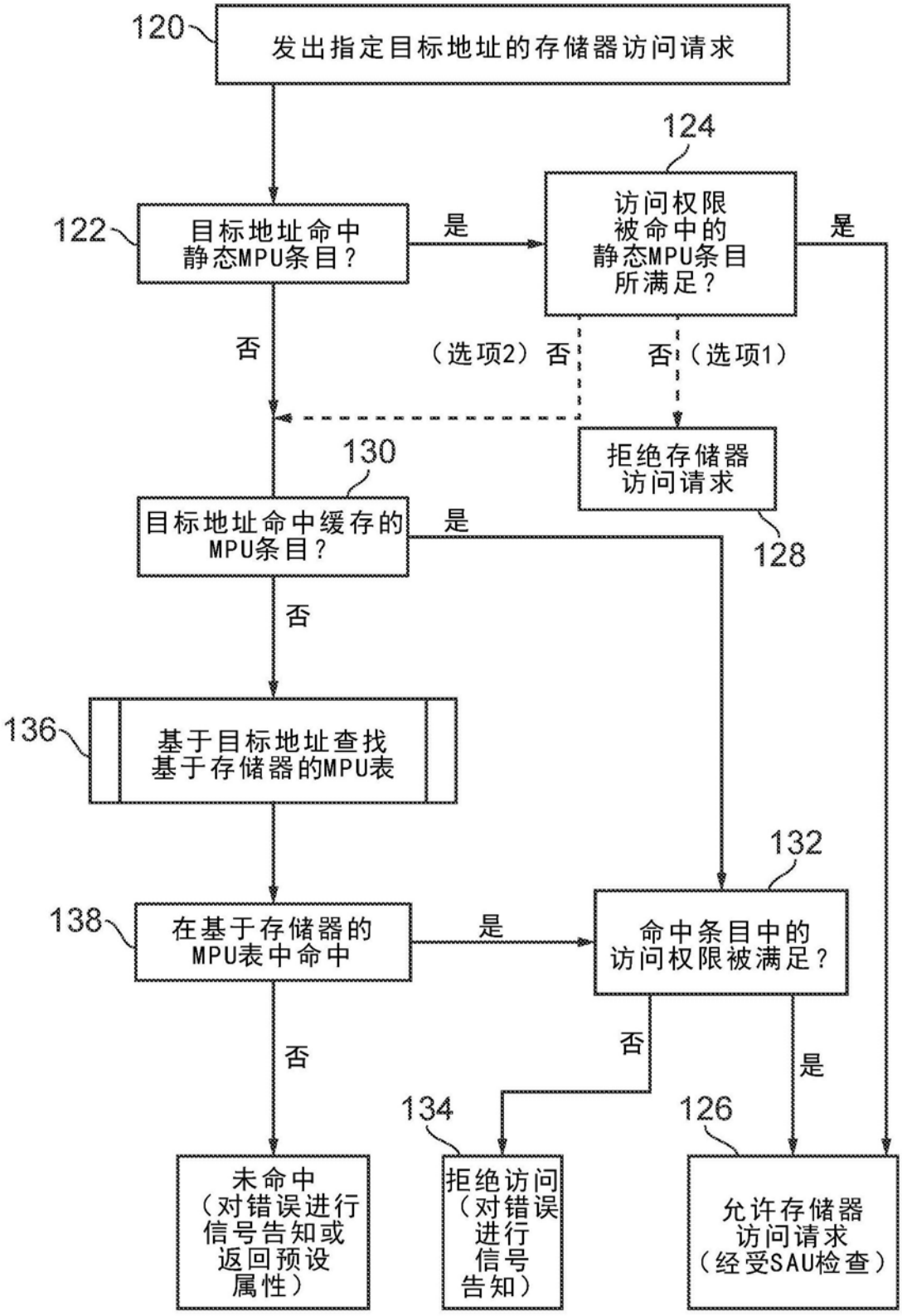

120
发出指定目标地址的存储器访问请求
124
访问权限
被命中的
静态MPU条目
所满足？
122
目标地址命中
静态MPU条目？
是
是
否
（选项2）否
否（选项1）
拒绝存储器
访问请求
128
130
目标地址命中缓存的
MPU条目？
是
否
136
基于目标地址查找
基于存储器的MPU表
132
138
在基于存储器的
MPU表中命中
是
命中条目中的
访问权限被满足？
否
否
是
134
126
未命中
（对错误进行
信号告知或
返回预设
属性）
拒绝访问
（对错误
进行
信号
告知）
允许存储器
访问请求
（经受SAU检查）

图6

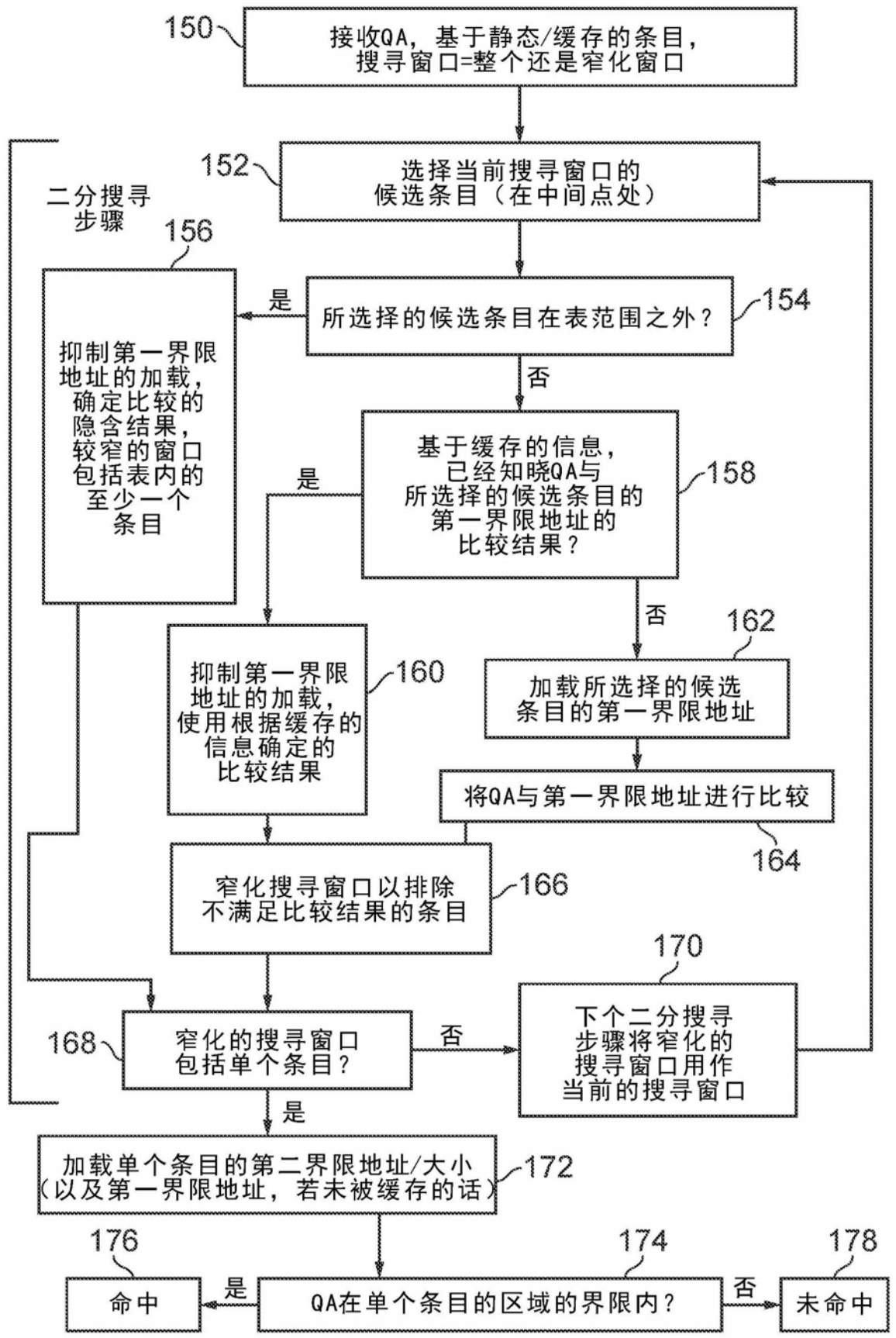

150
接收QA，基于静态/缓存的条目，
搜寻窗口=整个还是窄化窗口
152
选择当前搜寻窗口的
候选条目（在中间点处）
二分搜寻
步骤
156
抑制第一界限
地址的加载，
确定比较的
隐含结果，
较窄的窗口
包括表内的
至少一个
条目
是
所选择的候选条目在表范围之外？
154
否
基于缓存的信息，
已经知晓QA与
所选择的候选条目的
第一界限地址的
比较结果？
158
是
否
162
抑制第一界限
地址的加载，
使用根据缓存的
信息确定的
比较结果
160
加载所选择的候选
条目的第一界限地址
将QA与第一界限地址进行比较
164
窄化搜寻窗口以排除
不满足比较结果的条目
166
170
168
窄化的搜寻窗口
包括单个条目？
否
下个二分搜寻
步骤将窄化的
搜寻窗口用作
当前的搜寻窗口
是
加载单个条目的第二界限地址/大小
（以及第一界限地址，若未被缓存的话）
172
176
174
178
命中
是
QA在单个条目的区域的界限内？
否
未命中

图7

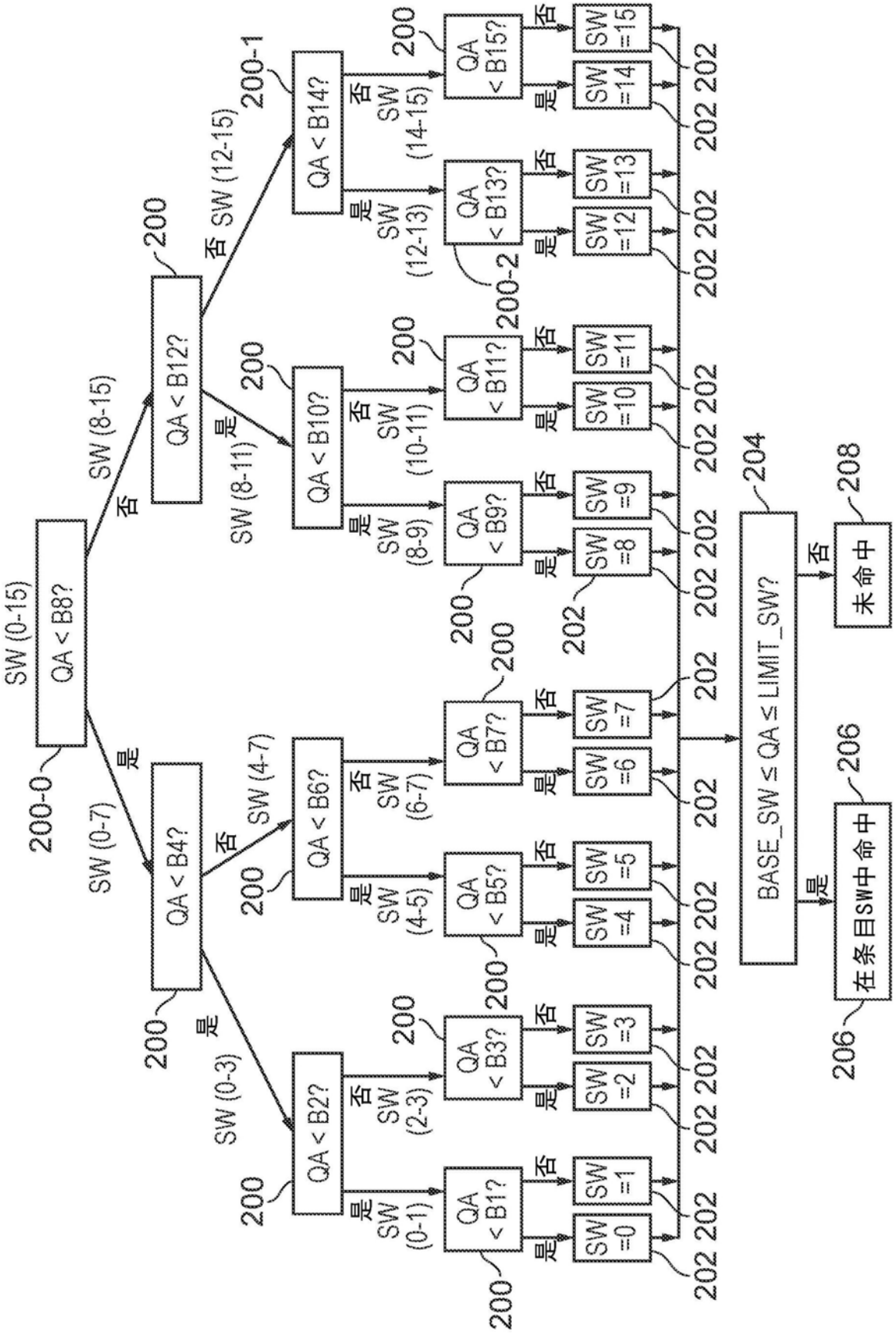

SW (0-15)
QA < B8?
200-0
SW (0-7)
是
否
SW (8-15)
QA < B4?
200
SW (0-3)
是
否
SW (4-7)
QA < B12?
200
SW (8-11)
是
否
SW (12-15)
QA < B2?
200
SW (0-1)
是
否
SW (2-3)
QA < B6?
200
SW (4-5)
是
否
SW (6-7)
QA < B10?
200
SW (8-9)
是
否
SW (10-11)
QA < B14?
200-1
SW (12-13)
是
否
SW (14-15)
QA < B1?
200
是
否
SW =0
SW =1
QA < B3?
200
是
否
SW =2
SW =3
QA < B5?
200
是
否
SW =4
SW =5
QA < B7?
200
是
否
SW =6
SW =7
QA < B9?
200
是
否
SW =8
SW =9
QA < B11?
200
是
否
SW =10
SW =11
QA < B13?
200-2
是
否
SW =12
SW =13
QA < B15?
200
是
否
SW =14
SW =15
202
BASE_SW ≤ QA ≤ LIMIT_SW?
204
是
否
在条目SW中命中
206
未命中
208

图8

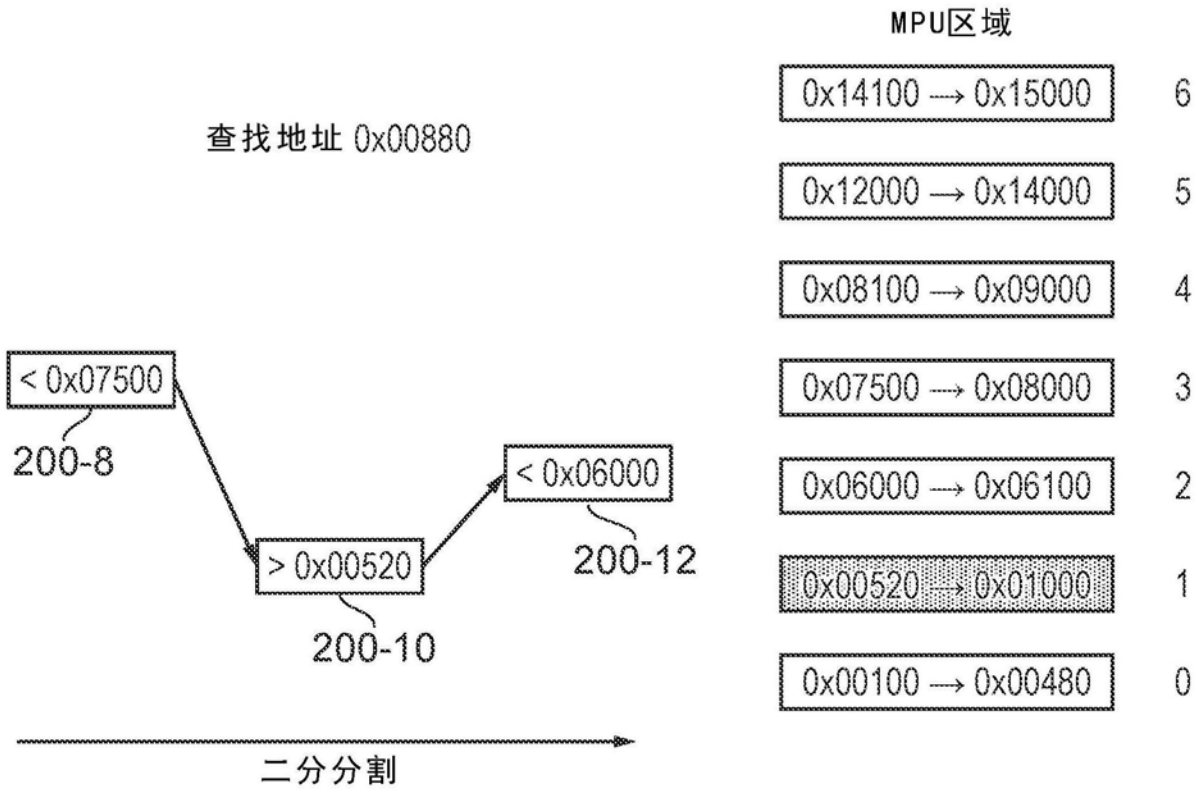

MPU区域
0x14100 → 0x15000
6
0x12000 → 0x14000
5
0x08100 → 0x09000
4
0x07500 → 0x08000
3
0x06000 → 0x06100
2
0x00520 → 0x01000
1
0x00100 → 0x00480
0
查找地址 0x00880
< 0x07500
200-8
> 0x00520
200-10
< 0x06000
200-12
二分分割

图9

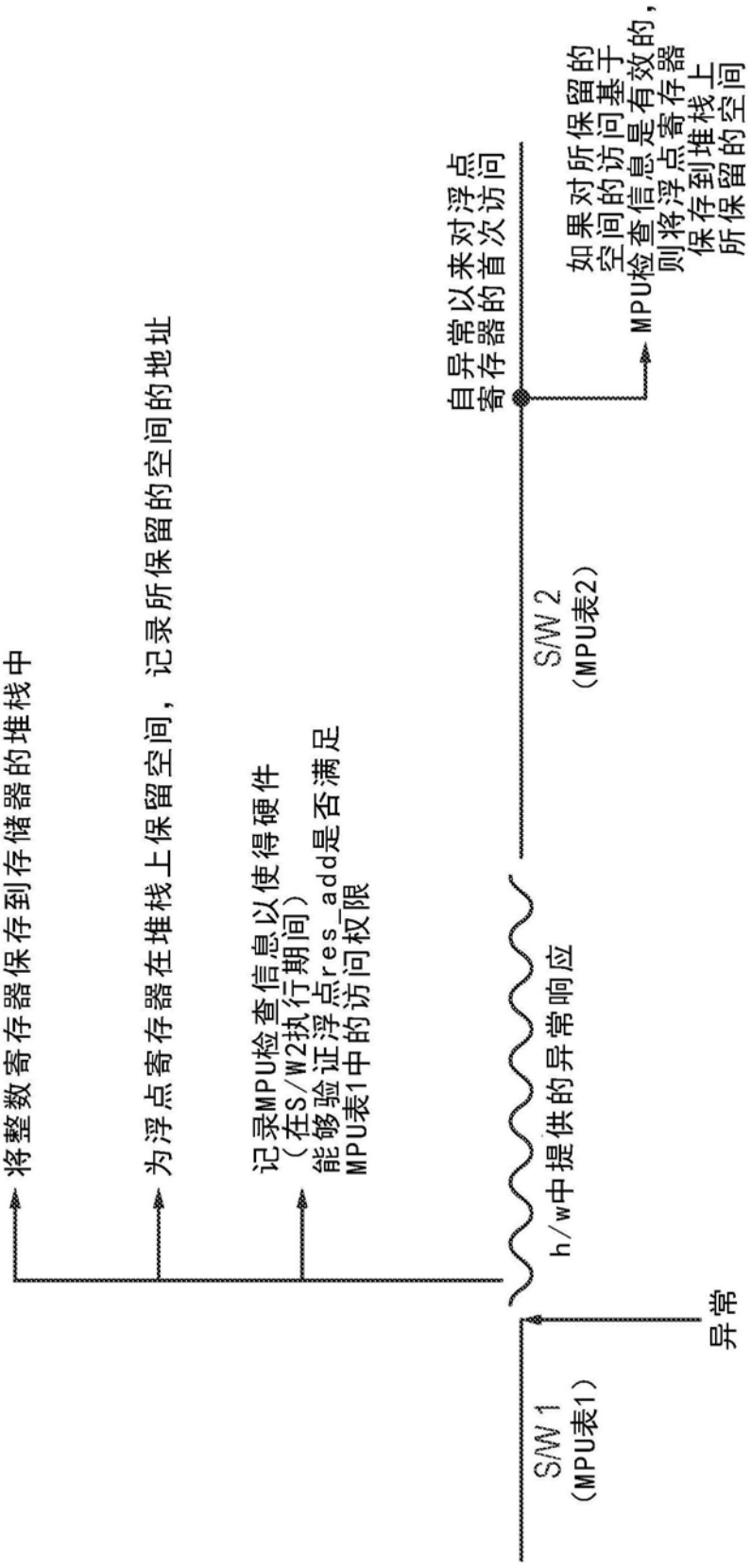

将整数寄存器保存到存储器的堆栈中
为浮点寄存器在堆栈上保留空间，记录所保留的空间的地址
记录MPU检查信息以使得硬件
（在S/W2执行期间）
能够验证浮点res_add是否满足
MPU表1中的访问权限
h/w中提供的异常响应
异常
S/W 1
（MPU表1）
S/W 2
（MPU表2）
自异常以来对浮点
寄存器的首次访问
如果对所保留的
空间的访问基于
MPU检查信息是有效的，
则将浮点寄存器
保存到堆栈上
所保留的空间

图10

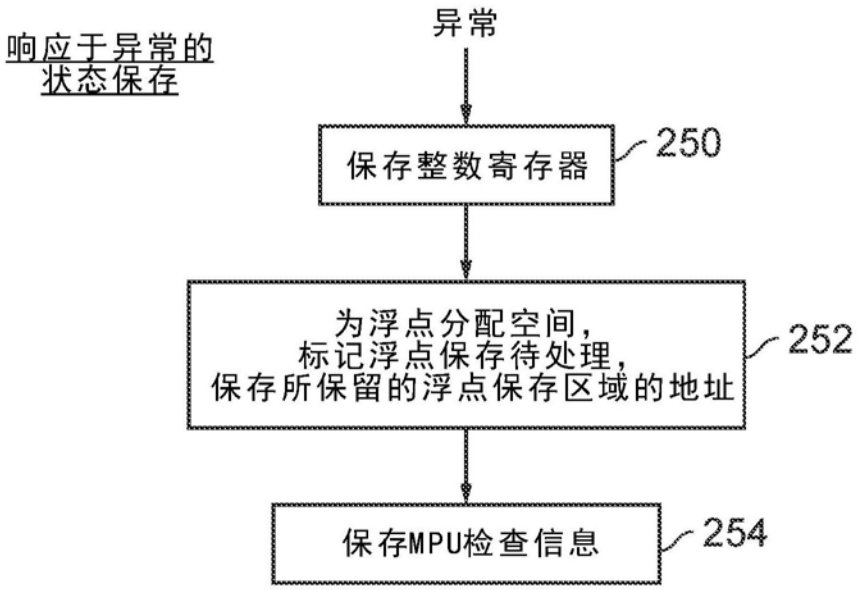

响应于异常的
状态保存
异常
保存整数寄存器
250
为浮点分配空间，
标记浮点保存待处理，
保存所保留的浮点保存区域的地址
252
保存MPU检查信息
254

图11A

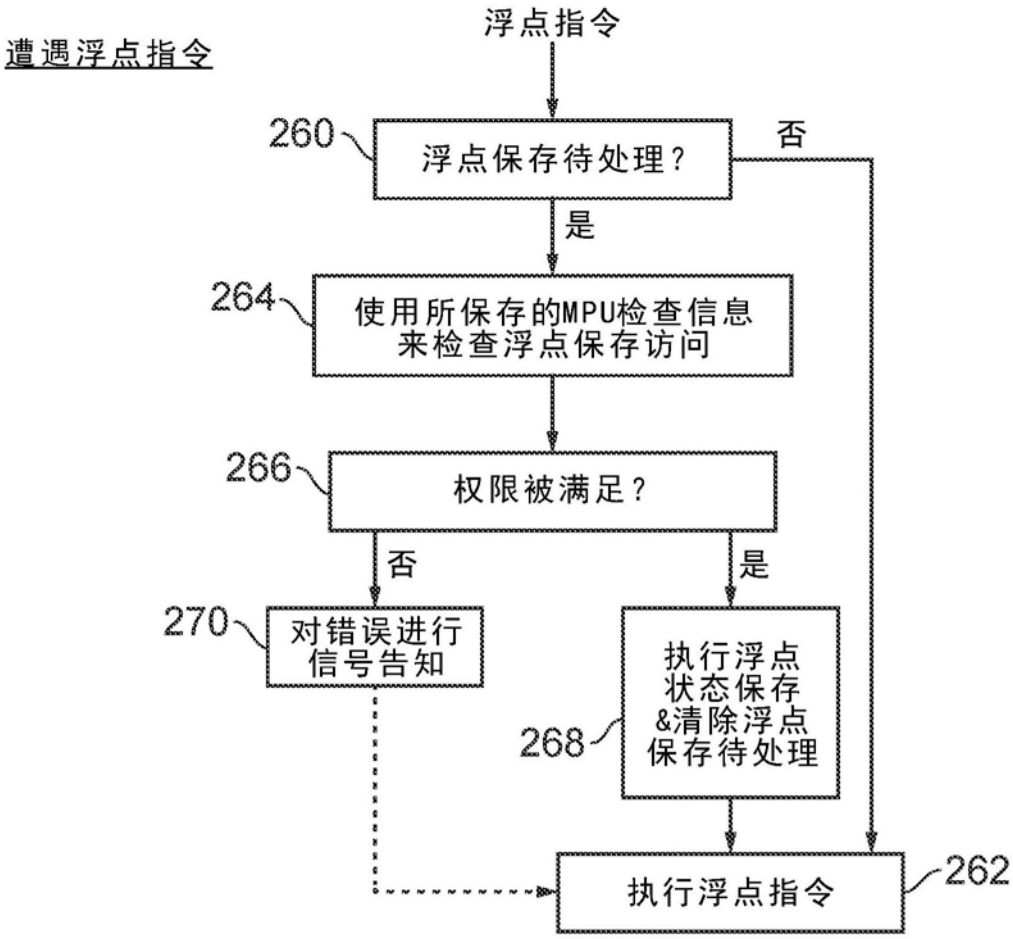

遭遇浮点指令
浮点指令
260
浮点保存待处理？
否
是
264
使用所保存的MPU检查信息
来检查浮点保存访问
266
权限被满足？
否
是
270
对错误进行
信号告知
执行浮点
状态保存
&清除浮点
保存待处理
268
执行浮点指令
262

图11B

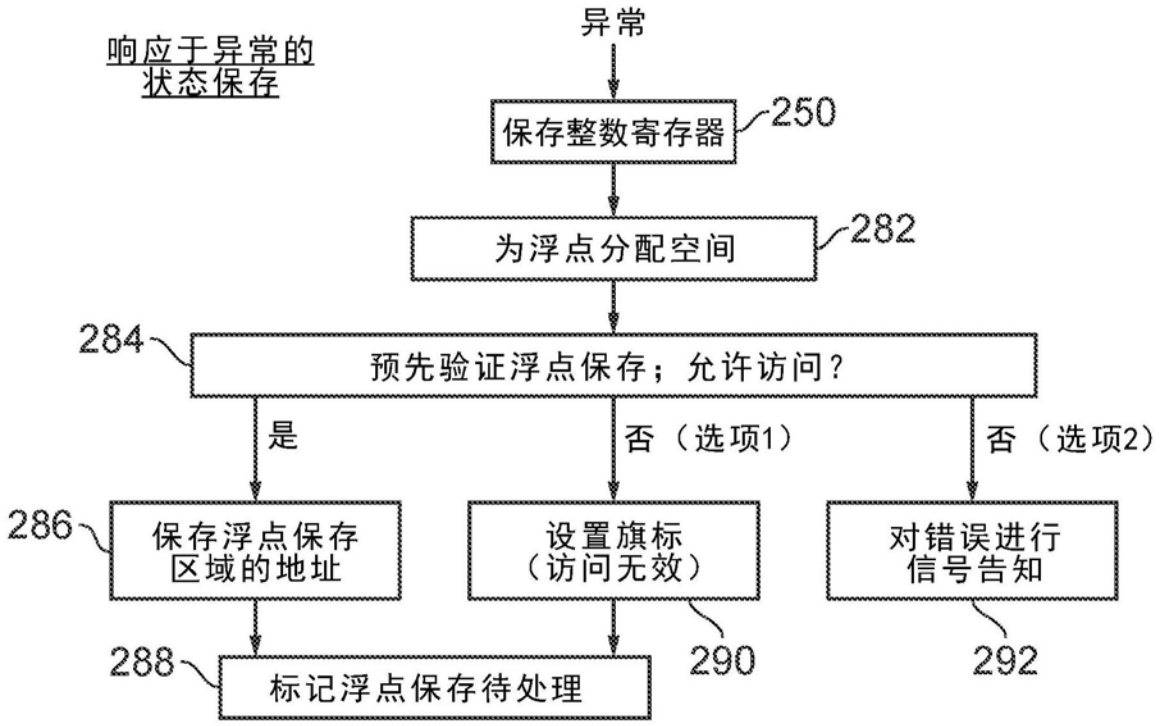

响应于异常的
状态保存
异常
保存整数寄存器
250
为浮点分配空间
282
284
预先验证浮点保存；允许访问？
是
否（选项1）
否（选项2）
286
保存浮点保存
区域的地址
设置旗标
（访问无效）
对错误进行
信号告知
290
292
288
标记浮点保存待处理

图12A

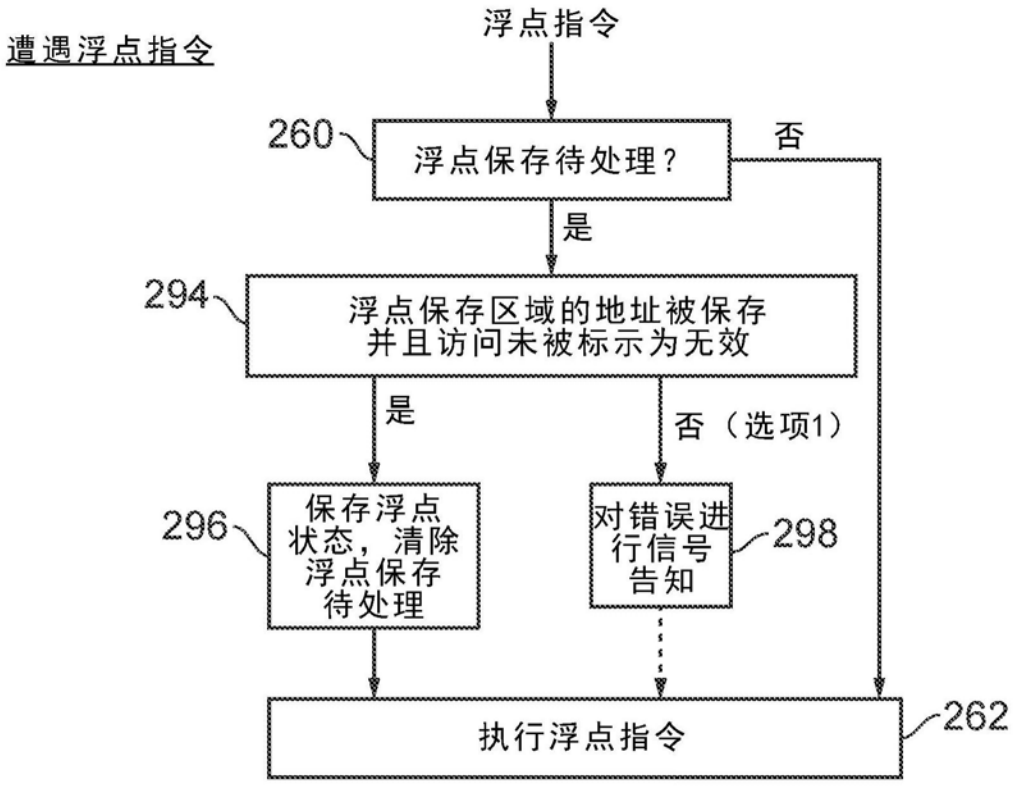

遭遇浮点指令
浮点指令
260
浮点保存待处理？
否
是
294
浮点保存区域的地址被保存
并且访问未被标示为无效
是
否（选项1）
296
保存浮点
状态，清除
浮点保存
待处理
对错误进
行信号
告知
298
执行浮点指令
262

图12B

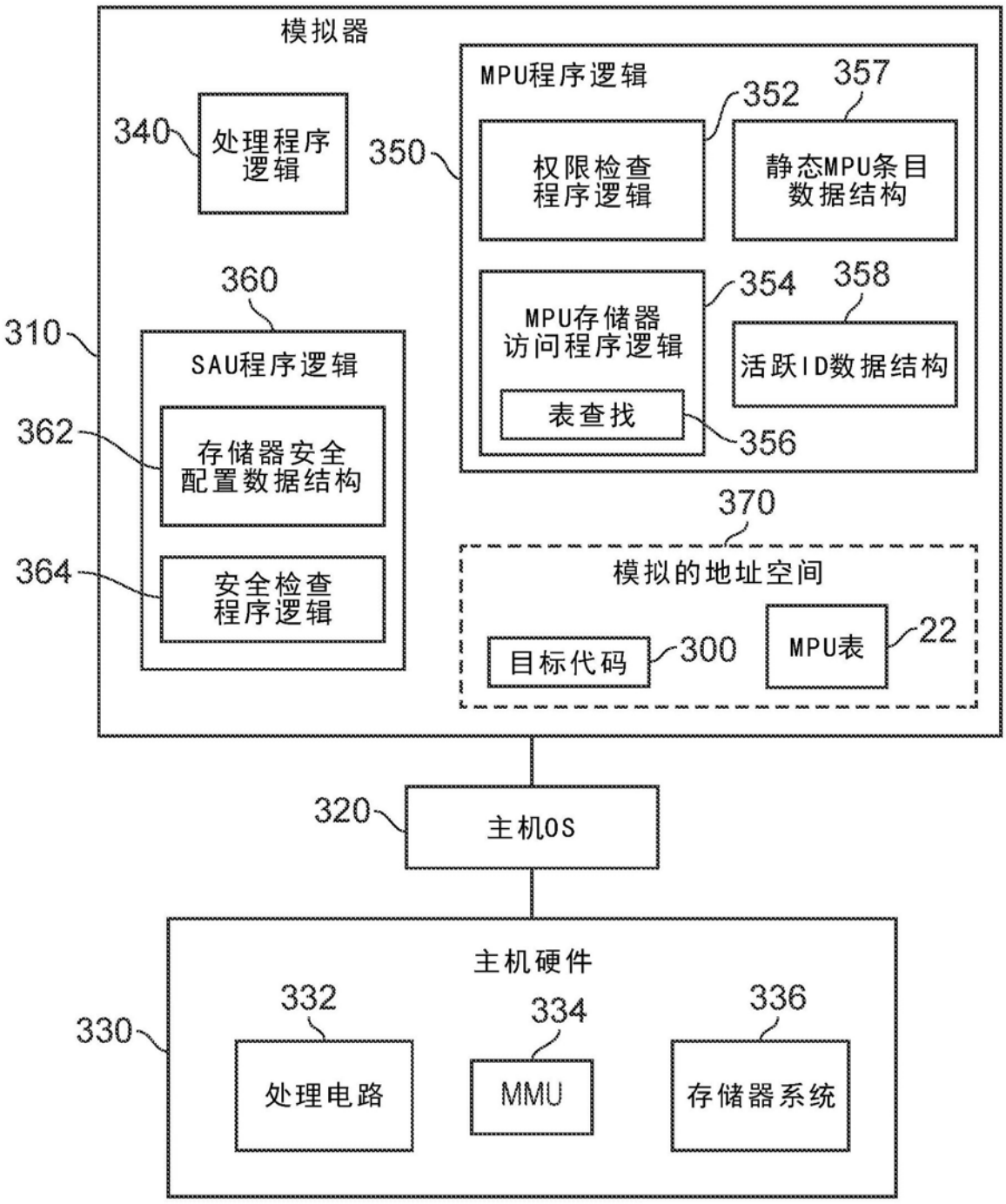

模拟器
MPU程序逻辑
357
352
340
处理程序逻辑
350
权限检查程序逻辑
静态MPU条目数据结构
360
354
358
310
SAU程序逻辑
MPU存储器访问程序逻辑
活跃ID数据结构
362
存储器安全配置数据结构
表查找
356
370
364
安全检查程序逻辑
模拟的地址空间
目标代码
300
MPU表
22
320
主机OS
主机硬件
332
334
336
330
处理电路
MMU
存储器系统

图13