[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 9/26 (2006.01)
G06F 9/34 (2006.01)
G06F 12/00 (2006.01)

[12] 发明专利申请公布说明书

[21] 申请号 200680001671.3

[43] 公开日 2009 年 8 月 19 日

[11] 公开号 CN 101512482A

[22] 申请日 2006.1.9
[21] 申请号 200680001671.3
[30] 优先权
[32] 2005. 2. 8 [33] US [31] 11/054,076
[86] 国际申请 PCT/US2006/000602 2006.1.9
[87] 国际公布 WO2006/086101 英 2006.8.17
[85] 进入国家阶段日期 2007.6.29
[71] 申请人 思科技术公司
地址 美国加利福尼亚州
[72] 发明人 威尔·伊瑟顿 厄尔·科亨
安迪·费戈哈特
唐纳德·E·斯特斯 约翰·威廉斯

[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
代理人 王 怡

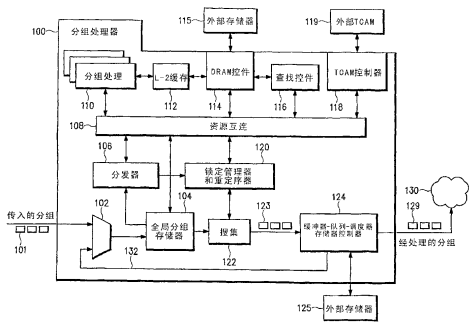
权利要求书 6 页 说明书 20 页 附图 19 页

[54] 发明名称

多线程分组处理体系结构

[57] 摘要

一种网络处理器具有许多新颖的特征，包括多线程处理器阵列、多轮回处理模型和具有硬件管理的分组存储装置的全局分组存储器(GPM)。 这些独特的特征允许网络处理器以高数据速率执行高接触型分组处理。 网络处理器还可利用基于堆栈的高级编程语言(例如 C 或 C + +)来编码。 这允许了将软件特征更迅速、更高质量地移植到网络处理器中。 当额外的处理特征被添加时，处理器性能也不会严重降低。 例如，通过将处理元件分派给不同的有限持续时间到达处理任务和可变持续时间主处理任务，可更智能地处理分组。 再循环路径在不同的到达和主处理任务之间移动分组。 其他新颖的硬件特征包括将协同处理器操作与多线程处理操作高效混合并提高了缓存亲和力的硬件体系结构。



1. 一种分组处理器，包括：

一个或多个处理器，其运行一个或多个线程，每个线程具有关联的线程标识符并且生成虚拟地址；

第一存储器映射级，其将所述虚拟地址映射到与不同资源相关联的地址值；以及

与所述不同资源中的一个或多个相关联的第二存储器映射级，其将所述地址值映射到所述资源中与所述关联线程标识符相对应的物理地址区域。

2. 如权利要求 1 所述的分组处理器，其中，所述第一存储器映射级包括与所述线程相关联的一个或多个转化缓冲器。

3. 如权利要求 1 所述的分组处理器，其中，所述第二存储器映射级包括分组句柄数据结构，该分组句柄数据结构使用所述线程标识符和所述地址值来识别与个体线程相关联的分组数据位置。

4. 如权利要求 2 所述的分组处理器，包括将所述虚拟地址映射到第一分组存储器资源的第一转化缓冲器条目和将所述虚拟地址映射到第二外部存储器资源的第二转化缓冲器条目。

5. 如权利要求 4 所述的分组处理器，包括将所述虚拟地址映射到所述第二外部存储器资源内的堆栈区域的第三转化缓冲器条目，并且其中所述第二存储器映射级随后将来自所述第三转化缓冲器条目的所述地址值映射到所述外部存储器中与所述线程标识符相对应的堆栈。

6. 如权利要求 1 所述的分组处理器，包括将分组分配给所述线程的分发器，以及搜集和组装被所述线程处理的分组的搜集机构。

7. 如权利要求 6 所述的分组处理器，包括维护分组句柄数据结构的全局分组存储器，该分组句柄数据结构被所述线程在处理所述分组时使用，并且在所述线程完成所述分组的处理后被所述搜集机构用于搜集和组装所述分组。

8. 如权利要求 1 所述的分组处理器，包括根据从相应的线程接收的线

程命令、地址值和线程标识符来访问和处理分组的协同处理器。

9. 如权利要求 8 所述的分组处理器，其中，一个或多个线程与所述协同处理器并行地处理相同或不同分组。

10. 一种分组处理器，包括：

一个或多个分组处理器元件，每个分组处理器元件操作一个或多个线程；

存储分组的分组存储器；

分发器，其根据所述分组所需的有限时间到达处理或可变时间主处理将所述分组存储器中的分组分发到所述线程。

11. 如权利要求 10 所述的分组处理器，包括排队系统，该排队系统在完成到达处理和主处理中的每一种之后接收所述分组。

12. 如权利要求 11 所述的分组处理器，包括再循环路径，该再循环路径在完成所述到达处理或所述主处理器之后将所述排队系统中的分组再循环回所述分组存储器。

13. 如权利要求 12 所述的分组处理器，其中，所述再循环路径将所述排队系统中的分组再循环回所述分组存储器，以进行一次或多次额外的主处理轮回。

14. 如权利要求 12 所述的分组处理器，其中，所述主处理部分地完成多播分组的分组复制，然后通过所述再循环路径发回所述多播分组以再一次轮回经过主处理以便继续所述分组复制。

15. 如权利要求 11 所述的分组处理器，其中，所述排队系统包括根和在完成到达或主处理之后存储分组的关联队列，所述根中的某些被配置为将所述关联队列中的分组发回所述分组存储器以便所述线程进行额外主处理的再循环根。

16. 如权利要求 15 所述的分组处理器，包括标识与不同根相关联的主线程的主分发群组，所述分发器优选地利用所述主分发群组来将分组分派给关联的主线程。

17. 如权利要求 16 所述的分组处理器，包括标识与不同根相关联的次线程的次分发群组，所述分发器在关联的主线程不可用时利用所述次分发

群组来将分组分派给关联的次线程。

18. 如权利要求 17 所述的分组处理器，其中，所述次线程在多于一个次分发群组中是部分重叠的。

19. 如权利要求 11 所述的分组处理器，包括搜集机构，在先前处理分组的线程处理新分组的同时，该搜集机构搜集所述分组存储器中的分组以便发送到所述排队系统。

20. 一种网络处理系统，包括：

一个或多个分组处理元件，每个分组处理元件操作一个或多个线程，每个线程具有关联的线程标识符；

访问与所述线程标识符相对应的不同存储器区域中的分组的存储器单元；

协同处理器，其接收来自所述线程的命令以处理所述分组，并随后利用所述线程标识符来独立地访问和处理所述存储器单元中的分组。

21. 如权利要求 20 所述的网络处理系统，包括操作向所述协同处理器发送命令以处理第一分组的第一线程的第一处理元件，所述第一线程随后等待直到所述协同处理器完成所述第一分组的处理，或者在所述协同处理器处理所述第一分组的同时继续处理所述第一分组的另一部分。

22. 如权利要求 21 所述的网络处理系统，包括所述第一处理元件中或另一个处理元件中的第二线程，该第二线程在所述协同处理器处理所述第二分组的同时处理所述存储器单元中的第二分组。

23. 如权利要求 20 所述的网络处理系统，其中，对于所述存储器单元中的相同存储器位置中的相同分组，所述线程有选择地将其自己的分组处理与所述协同处理器进行的分组处理相混合。

24. 如权利要求 20 所述的网络处理系统，包括资源互连，该资源互连将所述分组处理元件与所述存储器单元相耦合，并且并行地将所述协同处理器与所述存储器单元相耦合。

25. 如权利要求 20 所述的网络处理系统，其中，所述分组处理元件通过同一资源互连访问所述协同处理器和所述存储器单元。

26. 如权利要求 20 所述的网络处理系统，包括被一个或多个所述线程

访问的指令缓存和数据缓存。

27. 如权利要求 20 所述的网络处理系统，包括与个体线程相关联的个体数据缓存或个体指令缓存。

28. 如权利要求 20 所述的网络处理系统，包括：

第一地址转化缓冲器，其将线程虚拟地址映射到关联的资源中的地址值；以及

所述关联的资源中的第二存储器映射，其根据关联的线程标识符将所述地址值映射到所述资源中的物理存储器位置。

29. 如权利要求 20 所述的网络处理系统，包括将所述存储器单元中的分组分派给所述线程的分发器。

30. 如权利要求 29 所述的网络处理系统，其中，所述分发器根据与分组相关联的第一到达分组处理级和第二主处理级将所述分组分配给线程。

31. 如权利要求 30 所述的网络处理系统，包括接收首先被所述分发器分配给所述第一到达分组处理级的分组的一个或多个外部端口，以及将分组发回所述分发器以便分配给所述第二主处理级的再循环路径。

32. 如权利要求 31 所述的网络处理系统，包括排队系统，该排队系统在完成所述第一到达分组处理级和所述第二主处理级中的每一个之后接收分组。

33. 一种分组处理元件，包括：

操作一个或多个线程的分组处理元件；

与所述线程相对应的一个或多个转化缓冲器，所述转化缓冲器具有与不同资源或所述资源内的不同类型的数据相对应的不同条目；

包含由所述线程访问的缓存行的一个或多个缓存；以及

缓存控制器，其接收来自所述线程的标识所述转化缓冲器条目的缓存命令，并且对与标识出的转化缓冲器条目相对应的所有缓存行执行所述缓存命令。

34. 如权利要求 33 所述的分组处理元件，其中，所述缓存命令之一使得所述缓存控制器将与所述标识出的转化缓冲器条目相对应的所有缓存行的数据存储到主存储器中。

35. 如权利要求 33 所述的分组处理元件，其中，所述缓存命令之一使得所述缓存控制器将与所述标识出的转化缓冲器条目相对应的所有经修改的缓存行的数据存储到主存储器中。

36. 如权利要求 33 所述的分组处理元件，其中，所述缓存命令之一使得所述缓存控制器将与所述标识出的转化缓冲器条目相对应的经修改的缓存行的数据存储到主存储器中，然后使与所述标识出的转化缓冲器条目相对应的所有缓存行无效。

37. 如权利要求 33 所述的分组处理元件，其中，所述控制器接收标识所述转化缓冲器条目之一的单个缓存命令，将所述缓存中的每个标签与对应于所述标识出的转化缓冲器条目的物理地址范围相比较，并且对具有在所述物理地址范围内的标签的每个缓存行执行所述缓存命令。

38. 如权利要求 37 所述的分组处理元件，具有存储用于多个不同线程的指令的单个缓存。

39. 如权利要求 33 所述的分组处理元件，包括存储用于多个不同线程的分组数据的单个数据缓存阵列和用于每个线程的个体标签阵列，或者存储用于个体线程的分组数据的多个不同缓存。

40. 一种分组处理器，包括：

操作一个或多个线程的分组处理元件的阵列；以及

存储器系统，其包括用于存储分组的分组存储器和具有与所述分组存储器中的个体分组相对应的条目的分组句柄数据结构，所述分组和所述分组句柄数据结构在所述线程已完成对所述分组的处理后被维护在所述存储器系统中，并随后被用于后续的分组搜集和组装。

41. 如权利要求 40 所述的分组处理器，包括分发器，其在所述线程先前处理的分组和关联的分组句柄数据结构保持在所述存储器系统以便进行所述搜集和组装操作的同时，向所述线程分派新的分组以便处理。

42. 如权利要求 41 所述的分组处理器，其中，所述分发器根据所述分组所需的相对有限的到达处理时间和可变的主处理时间的类型，将所述分组分配给不同的线程。

43. 如权利要求 40 所述的分组处理器，包括搜集机构，该搜集机构使

用所述线程先前释放的分组的分组句柄数据结构来独立地搜集和组装释放的分组以便排队。

44. 如权利要求 40 所述的分组处理器，包括排队系统，该排队系统包括再循环路径，该再循环路径将排队的分组再循环回所述存储器系统以便所述线程进行额外处理。

45. 如权利要求 44 所述的分组处理器，其中，所述排队系统包括具有关联的根的队列，所述根中的某些被配置为经由所述再循环路径将分组发回所述存储器系统。

46. 如权利要求 40 所述的分组处理器，包括根据所述线程发送的命令、线程标识符和地址值来自治地处理所述分组的协同处理器。

47. 一种分组处理器，包括：

用于处理分组的一个或多个分组处理元件；

排队系统，用于在所述分组被所述分组处理元件处理之后存储所述分组；以及

分发器，其将接收自一个或多个外部端口和所述排队系统的分组分配给所述分组处理元件。

48. 如权利要求 47 所述的分组处理器，其中，所述分发器将接收自所述外部端口的分组分配给提供到达处理操作的分组处理元件，并将接收自所述排队系统的分组分配给提供主处理的分组处理元件。

49. 如权利要求 48 所述的分组处理器，包括再循环路径，该再循环路径在完成所述到达处理或所述主处理之后将所述排队系统中的分组再循环回所述分发器以便重新分配回所述分组处理元件。

50. 如权利要求 48 所述的分组处理器，其中，所述到达处理在所述分组处理元件中具有相对有限的处理时间，所述主处理在所述分组处理中具有可变处理时间。

51. 如权利要求 47 所述的分组处理器，其中，所述排队系统包括根和存储所述分组的关联的队列，所述根中的某些被配置为将所述关联队列中的分组发回所述分发器以便重新分配给所述分组处理元件的再循环根。

多线程分组处理体系结构

技术领域

本发明的一个实施例涉及通信和计算机系统，更具体而言涉及路由器、分组交换系统和其他网络分组处理设备。

背景技术

通信工业正在迅速变化以适应新兴的技术和不断增长的客户需求。对于新的网络应用和更高性能的需求正要求通信网络以更快的速度（例如更高的带宽）工作。许多通信提供商使用分组交换技术来实现这些目标。例如，使用支持因特网协议（IP）的分组交换和路由技术。

网络处理器几年来已被用于分组交换网络中，以中等到较高的分组处理速率提供划算的“高接触型（high touch）”分组服务。网络处理器经常有专门的微引擎用于分组处理应用。但是，网络处理器一般是难以编程的，尤其难以被编程以新特征。处理器还经常在启用额外的软件特征时陷入严峻的性能处境。

还存在在单个芯片上提供多个处理器的网络处理器体系结构。这些多处理器设备可包括分组处理辅助装置和专门的接口。这些多处理器体系结构通常是可以用 C 编程语言来编码的通用设备。但是，这些体系结构的通用性往往会限制它们的可缩放性和吞吐量。

一些网络处理器局限于 C 编程语言的非 ANSI 子集。由于缺乏清洁的堆栈模型，这些处理器不能被认为是通用的。

其他网络处理器体系结构使用处理器流水线，并且还可包括用于分组处理和其他处理器间通信的特殊硬件辅助装置。但是流水线处理器系统经常是非对称的，也就是说不是所有处理器都具有对所有资源的相同访问权限。

因此，需要一种具有更高的分组处理能力、可缩放性和工作灵活性的

网络处理器。本发明解决了这一问题以及与现有技术相关联的其他问题。

发明内容

根据本发明的网络处理器具有许多新颖的特征，包括多线程处理器阵列、多轮回处理模型和具有硬件管理的分组存储装置的全局分组存储器（GPM）。这些独特的特征允许网络处理器以高数据速率执行高接触型分组处理。网络处理器还可利用基于堆栈的高级编程语言（例如 C 或 C++）来编码。这允许了将软件特征更迅速、更高质量地移植到网络处理器中。

当额外的处理特征被添加时，处理器性能也不会严重降低。例如，通过将处理元件分派给不同的有限持续时间到达处理任务和可变持续时间主处理任务，可更智能地处理分组。再循环路径在不同的到达和主处理任务之间移动分组。其他新颖的硬件特征包括将协同处理器操作与多线程处理操作高效混合并提高了缓存亲和力的硬件体系结构。

从以下参考附图对本发明的优选实施例的详细描述中，可以更容易清楚本发明的前述和其他目的、特征和优点。

附图说明

图 1 是多线程分组处理器的框图。

图 2 是描述图 1 中的分组处理器的某些操作的流程图。

图 3 和 4 是示出分组处理器中的处理元件之间的通信的图。

图 5 是分组处理器中的全局分组存储器所使用的数据结构。

图 6 是分组处理器中使用的分组处理元件（PPE）的详细框图。

图 7 和 8 是示出 PPE 中的不同线程如何被分配分组的框图。

图 9A 和 9B 是示出到达和主分组处理任务如何被分配给不同线程的图。

图 10A、10B、11A 和 11B 示出分组处理器如何利用再循环来更智能地处理分组。

图 12A 和 12B 示出分发群组如何被用于提高缓存亲和力。

图 13 和 14 示出工作在分组处理器中的双层存储器映射系统。

图15示出协同处理器分组处理如何与多线程分组处理操作相混合。

图16和17示出如何在分组处理器中提高缓存一致性（cache coherency）。

具体实施方式

图1是多线程分组处理器100的框图。分组101被分组处理器100所接收，并且一般经由复用器102被存储在全局分组存储器（GPM）104中。在分组被接收后，GPM 104构建一个关联的分组句柄数据结构（图5），然后将分组加入到由锁定管理器和重定序器120操作的流程锁定（flow lock）队列上。在接收到从锁定管理器120返回的答复之后，GPM 104指示分发器106将分组101分配给分组处理元件（PPE）110。

PPE 110通过资源互连108处理GPM 104中的分组。PPE 110还可使用第二层（L2）缓存112、动态随机访问存储器（DRAM）控件114和查找控件116来访问外部存储器115。外部三元内容可寻址存储器（TCAM）119也可被PPE 110通过资源互连108和TCAM控制器118访问。在一个实施例中，PPE 110是多线程的。但是，下文描述的某些特征可由具有或不具有多线程能力的任何通用处理单元所执行。

PPE 110在其完成处理分组时通知锁定管理器120。然后PPE 110就可自由地开始处理其他分组。在被PPE 110处理之后，分组继续驻留在GPM 104中，并且可以按分散的非毗邻方式存储在GPM 104中。搜集机构122负责搜集分组的分散部分并将其再组装在一起。锁定管理器120与搜集机构122一起工作以确定组装的分组123被从GPM 104发送到缓冲器-队列-调度器（BQS）存储器控制器124的最终顺序。BQS 124对分组进行排队、调度和出队，以使PPE 110能够免除这一耗时的任务。外部存储器125被BQS 124用作分组缓冲器，用于在不同的到达和主处理操作之间存储分组，等等。再循环路径132被BQS 124用于将分组再循环回GPM 104以供PPE 110进一步处理。

各种专门的分组处理辅助装置，例如转发信息数据库（FIB）查找、TCAM访问控制器118、对存储器的原子操作、策略管理器（policer）、

加权随机早期检测（WRED）、散列和模数等等，也使得分组处理器 100 能够提供更高的性能水平。分组处理辅助装置还提供对已知数据结构的硬件原子更新，以允许对代表经过网络处理器的大带宽流的结构进行高性能更新。

分组处理器 100 中的资源是指能够被 PPE 110 访问的任何不同的功能元件。例如，L-2 缓存 112、外部存储器 115、外部 TCAM 119、GPM 104、协同处理器 634（图 15）等等都被认为是资源。

图 2 更详细地示出了分组处理器 100 的一个实施例的一般操作。在块 142 中，分组 101 被接收并存储在 GPM 104 中。在分组被接收之后，GPM 104 发出与分组所属的流（例如在其上接收到分组的接口）相对应的锁定请求。在另一实施例中，可以基于分组内容来识别流。

在块 144 中，分发器 106 识别用于分配给分组的线程。在锁定请求被确认回 GPM 104 之后，分发器 106 将分组分派通知给该线程。在块 146 中，分派的线程从 GPM 104 检索分组的相关部分（例如头部以及可能的其他字段），并且处理此信息和/或其他信息来识别与分组相关联的流程/锁定（如果存在的话）。然后线程继续处理分组。

根据判决块 148 中所确定的，如果要执行转换操作，则线程在块 150 中将转换指令关联/附加到当前的锁定请求。在当前的锁定请求被获取时，例如当相应的转换标识符到达相应锁定队列的头部时，锁定管理器 120 执行（或者使另一机构执行）将当前锁定转换成新锁定的指令，然后释放当前锁定。如果需要额外的锁定转换，则块 152 重复块 150 中的操作。

在块 154 中，当线程完成对分组的处理时，线程向锁定请求附加搜集指令。在判决块 156 中，锁定管理器 120 等待与分组相关联的分组句柄（handle）到达锁定队列的头部。锁定管理器 120 随后在块 158 中指示搜集机构 122 组装分组并将组装的分组 123（图 1）转发到 BQS 124。锁定管理器 120 随后释放锁定和分组句柄。

图 3 示出了分组处理器 100 中的初始操作和消息传递。该图中的操作按垂直顺序进行，从顶部开始向底部进展。多个 PPE 110 一般同时处理分组。同一个 PPE 110 也可利用不同的线程同时处理多个分组。从而，下文

描述的某些操作的顺序可能取决于流程锁定获取的标识符所需的时间、分组处理操作等等而不同。

特定分组被接收（210）并被存储（211）到 GPM 104 中。标识实际分组数据被存储在 GPM 104 中何处的分组句柄数据结构被分配。分组句柄数据结构将在下文于图 5 中更详细描述。与分组句柄数据结构相对应的分组句柄被传输（212）到分发器 106。分发器 106 在目前有空闲线程（free thread）可用的情况下或者在有空闲线程变得可用之后分配（213）一个空闲线程，并且向 GPM 104 标识（214）该线程。

为了维护接收分组的队列，GPM 104 向流程锁定管理器 120 发送流程锁定请求（215）。流程锁定管理器 120 对分组执行（216）锁定请求，并且通知（217）GPM 104，或者还可能通知分发器 106。在锁定请求被确认之后，分发器 106 于是就被允许通知（218）分派的线程开始处理分组。在一个实施例中，通知（218）还用来向线程确认对前一分组的处理的完成。在另一实施例中，锁定确认 217 被发送到 GPM 104，GPM 104 随后指示分发器 106 通知线程。

线程请求（219）并接收（221）来自 GPM 104 的与分组相对应的分组头部以及可能的其他字段和/或信息。在一个实施例中，GPM 104 基于图 5 中描述的分组句柄和线程 ID 来检索（220）该信息。

基于从 GPM 104 接收的信息，线程对分组进行分类（222），以识别可能存在的与子流相对应的要转换的额外锁定。线程向流程锁定管理器 120 提交（223）转换请求。流程锁定管理器 120 识别（224）与分组相对应的锁定标识符并添加转换指令。流程锁定管理器 120 随后向线程确认（225）转换请求已被添加。线程继续处理（226）分组。在当前的流程锁定被利用锁定标识符获取（227）时，其附加的指令被执行（227），这包括转换到新的锁定并释放先前的当前锁定。当然，这只是分组处理器 100 可执行的处理的一个特定部分的一个示例。

图 4 示出了可用于完成分组处理的一个场景，例如用于搜集分组并将其发送到 BQS 124（图 1）的操作。线程继续对分组的处理（226）。在完成分组处理之后，线程向流程锁定管理器 120 发送指令（231），例如：

分组搜集、将流程锁定转换为空（即不转换）以及释放流程锁定。流程锁定管理器 120 查找（232）与分组相对应的流程锁定标识符，并且作为响应，通知（233）分发器 106 释放线程。

分发器 106 释放线程以便它能够开始处理另一分组。取决于流量负载，线程可能被立即分派以另一分组，在刚才处理的分组被构建和/或发送之后（例如在搜集操作被执行时）被分派以另一分组，或者在当前处理的分组被实际构建和/或发送之后被分派以另一分组。任何之后分派的分组可以位于 GPM 104 中的相同或不同位置。

线程可能不被分派以全新的分发的分组。或者，线程可能保存当前分组数据并根据当前分组数据生成新的分组以便进行多播或者分段操作。

当与分组相对应的锁定标识符被获取（234）时，流程锁定管理器 120 向搜集机构 122 发布（235）搜集命令，该命令包括与分组相对应的分组句柄。搜集机构 122 获得（例如请求并接收）分组句柄数据结构的拷贝，随后释放（236）分组句柄和分组句柄数据结构。

搜集请求被搜集机构 122 入队（237）。当搜集请求被服务时（例如处于搜集队列的头部），实际的分组数据被请求（238）并从 GPM 104 接收（239）。分组随后被构建并发送（240）到 BQS 124，并且在 GPM 104 内分组数据空间被释放。

在一个实施例中使用单个搜集队列，而在另一个实施例中使用多个搜集队列。一般来说，多个搜集队列将通过一个或多个特性来区分，例如流量的优先级和/或类型、服务质量（QoS）、调度信息等等。

图 5 示出了 GPM 104 中用于存储和标识分组数据的数据结构的一个示例。分组句柄数据结构 304A-304N 一般是利用分组句柄 302A-302N 来访问的，所述分组句柄 302A-302N 可以是指针、偏移量值或者其他参考或地址值。在一个实施例中，分组句柄数据结构 304A-304N 被线程 402（图 6）用来访问分组存储器 311 中的不同的数据段 312 和 314。某些数据段 312 可包含实际的分组数据，而其他数据段 312 可包含与分组相关联的控制数据。

对于不同的实施例分组句柄数据结构 304 可以是不同的，但是它一般

包括诸如线程标识符（线程 ID）306 这样的描述符以及一个或多个指针 308 和 310。例如，GPM 开始位置指针 308 指向与特定分组（例如分组 1）相关联的第一数据段 312A。第一数据段 312A 可包含分组 1 的控制信息。GPM 结束位置指针 310 指向分组 1 的最末数据段 312D。类似地，第二分组句柄数据结构 304B 包括线程 ID 306 和指向分组存储器 311 中与另一分组（分组 2）相关联的其他数据段 314 的指针 308 和 310。

与特定分组相关联的数据段 312 可能分散在分组存储器 311 中的不同的非毗邻位置。一个单独的映射表（未示出）可包含将第一分组的不同数据段 312 彼此链接起来的链接 316。该映射表包括将另一个不同分组的数据段 314 彼此链接起来的其他链接，例如链接 317。

分组句柄数据结构 304 可以可选地包括标识分组存储器 311 中最近被访问的数据段的一个或多个动态缓存指针 320。例如，分组 1 的分组句柄 302A 中的地址偏移量 Y 可能访问了分组存储器 311 中的相应的数据段 312C。GPM 311 将偏移量值 Y 和数据段 312C 的相应物理地址写到分组句柄数据结构 304A 中的动态缓存指针 320 之一中。

之后的分组句柄 302A 可包括动态缓存指针 320 中与地址偏移量 Y 接近的地址偏移量。GPM 104 于是可利用动态缓存指针 320 直接跳到数据段 312C。如果识别出的数据段 312C 不包含与分组句柄 302A 中的地址偏移量相对应的分组数据，则 GPM 104 可以从指针 316C 开始，从而链接到正确的数据段 312。这与必须从第一分组数据段 312A 开始并随后顺序跳到每个后继的链接 316 直到定位到正确数据段 312 的情形相比要更快。

在一个实施例中，GPM 104 识别具有最接近但不小于分组句柄 302 中的偏移量地址值的地址偏移量值的动态缓存指针 320。GPM 104 从识别出的数据段 312C 的链接指针 316C 开始，以链接到正确的数据段。

在另一个实施例中，GPM 104 可包括反向指针 312 和 314。在该实施例中，GPM 104 可识别与分组句柄 302 中的偏移量值绝对最接近的动态缓存指针 320，而不论识别出的动态缓存指针 320 是在分组句柄值之上还是之下。GPM 104 随后可根据需要在数据段顺序中向前跳或向后跳。

即使 PPE 110 已经完成了对分组的主动处理，分组句柄数据结构 304

和分组存储器 311 中的数据段 312 和 314 仍继续驻留在 GPM 104 中（占有）。分组句柄数据结构 304 和关联的数据段 312 和 312 一般在 GPM 104 中保持有效，直到分组数据被传送到 BQS 124（图 1）。这允许了除 PPE 110 之外的资源对 GPM 104 中的分组数据执行任务。

对称处理

对称处理允许了在分组处理器 100 中工作的公共软件在任何 PPE 110 上运行任何线程。对于任何特定的 PPE 110 或线程都不需要处理专门化。从而，每当线程完成对分组的处理时，线程可被分发器 106 分派以任何新的分组，并且执行任何必要的分组处理任务。分组处理器 100 的另一个重要的特征是线程可以完成对某个分组的处理、被指派以新的分组并且开始处理该新分组，而不必等待先前处理的分组被输出到网络。例如，线程不必等待先前处理的分组被搜集和发送到 BQS 124（图 1），也不必等待分组被 BQS 124 从分组处理器 100 输出。

为了进一步说明，图 6 示出了 PPE 110 之一的更详细的图。中央处理单元 400 可操作多个线程 402。在该实施例中，每个线程 402 具有关联的数据缓存（DCACHE）标签和缓存控制器 404A，并且共享一个 DCACHE 数据阵列 404B。线程还共享同一个指令缓存（ICACHE）406。也可能有其他缓存配置，其中线程 402 都访问同一个 DCACHE 404，或者每个线程具有单独的 ICACHE 406。DCACHE 404 和 ICACHE 406 都可通过资源互连 108 访问外部存储器 115 和 GPM 104。在一个实施例中，ICACHE 112 还可通过 L-2 缓存 112 访问外部存储器 115，而 DCACHE 可直接访问外部存储器 115。当然，也可能有其他存储器配置，其中 DCACHE 404 通过 L-2 缓存 112 访问外部存储器 115，或者 ICACHE 406 直接访问外部存储器 115。

多线程 PPE 110 通过隐藏等待访问缓慢资源的等待时间来提高吞吐量。资源互连 108 向所有 PPE 110 提供对图 1 所示的所有资源的统一访问。从而，所有 PPE 110 和所有线程 402 具有对任何分组执行任何任务的相等的能力。PPE 110 各自支持一个堆栈模型并且具有一个转化后备缓冲器（Translation Look-aside Buffer，TLB）（图 16）以用于存储器映射。

图 7 示出了各自操作多个线程 402 的 PPE 110 的阵列。为了简便，下文中将分发器 106 和锁定管理器 120 总地称为控制器 410。到达分组被加载到 GPM 104 中，并且被排队以等待线程 402。控制器 403 发送指示线程 402A 开始处理第一分组 412A 的分配消息 414。同时或者在处理分组 412A 期间的某个时刻，控制器 410 可发送指示线程 402B 开始处理另一分组 412B 的另一分配消息 416。

当对分组 412A 的处理完成时，线程 402A 向 GPM 104 和控制器 410 发送通知 418。类似地，当对第二分组 412B 的处理完成时，线程 402B 向 GPM 104 和控制器 410 发回通知 420。应当理解，线程 402A 或线程 402B 都可首先完成对分配给其的分组的处理。如果必要，关联的分组句柄数据结构 304（图 5）上的分组 412A 和 412B 保持在 GPM 104 中，以便进行其他的排序、定序或线程处理操作。

参考图 8，在接收到分别来自线程 402A 或 402B 的通知 418 或 420（图 7）中的任何一个之后，并且在流程锁定被获取之后，控制器 410 向搜集机构 122 发送搜集指令 426，以对分组 412A 和/或 412B 开始搜集操作。同时，控制器 410 可向线程 402A 和/或 402B 分配另一分组。例如，分配指令 422 指示线程 402A 开始处理新的分组 412C，而分配指令 424 指示线程 402B 开始处理新的分组 412D。

在具有多个线程的其他网络处理单元中，流程锁定机构的缺乏阻碍了线程间的真正对称的处理并行化，从而要求流水线，或者导致对处理资源的利用不充分。但是，分组处理器 100 中的 PPE 110 和线程 402 在通知锁定管理器 120 对前一分组的处理完成之后可立即开始处理新的分组。这允许了 PPE 110 和关联的线程在先前处理的分组仍在 GPM 104 中被排队和搜集的同时或者在分组仍在 BQS 124 中被排队的同时开始处理其他分组。从而，PPE 110 和线程 402 仅受到实际处理分组所需的时间量的限制，而不必等待分组被调度或完成输入或输出排队。

分组再循环

再简要地返回参考图 1，处理分组所需的时间量可能是不同的。例如，分组可能在不同的链路上乱序到达或者作为片段到达。在等待其他分

组或分组片段到达的同时，分组或分组片段 101 可能必须被缓冲。在分组确实全部到达之后处理分组所需的时间量也可能取决于所启用的特征集而有所不同。例如，不同的处理特征，例如安全性处理或 QoS 处理，可能要求不同的处理时间量。

在这一可变时间处理期间的很长的等待时间时段可能导致分组队列中的阻塞，并最终导致分组丢弃。某些被丢弃的分组可能是用于维护网络链路的控制分组。其他被丢弃的分组可能影响分组优化级区分。例如，某些被丢弃的分组可能具有比其他分组更高的服务质量值。不幸的是，到达的分组可能在分组处理器有机会考虑关联的控制或 QoS 信息之前就被无差别地丢弃。

利用图 1 中的再循环路径 132 消除、减小或简化了这些问题中的某些。这种多轮回（multi-pass）处理特征支持一个确定性的“到达处理”轮回，该轮回执行最前面的时间有限的分组处理操作。然后可以针对可变运行时间高接触型处理执行第二“主处理”轮回。这种多轮回处理还增强了其他操作，例如提供了更精致的重组、多播等等。

为了进一步说明，图 9 示出了分成到达处理 452 和主处理 458 的整个 PPE 分组处理。到达处理 452 可以限于某些相对时间有限的处理工作，而到达处理 452 可执行可变时间“高接触型”处理操作。在到达处理 452 和主处理 458 期间，分组被存储在 GPM 104 中。第 1 级排队 456 和第 2 级排队由 BQS 124 提供。例如，BQS 124 中的第 1 级排队 456 在完成到达处理 452 之后将分组发回 GPM 104。BQS 124 中的第 2 级排队 462 用于将分组发回 GPM 104 以便进行额外的主处理 458 或者将分组发出到网络上。

如上所述，分发器 106 向 PPE 110 中的不同线程 402 分配到达分组处理和主分组处理任务。例如，分发器 106 可向某个线程子集 402A 发送分配命令 464 以便进行到达处理 452。类似地，分发器 106 可向另一线程子集 402B 发送分配命令 466 以便进行主分组处理 458。

在一个场景中，主处理 458 可能变得拥塞（瓶颈化），从而第 2 级排队 462 可能开始阻塞。如果必要，关于队列长度的信息或者在到达处理 452 期间确定的其他分组优先级区分信息可被用于作出更智能的分组丢弃

判决。从而，分组处理器 100 可以避免丢弃高优先级分组、控制流量等等。这允许了分组处理器 100 提供更多的服务，而不会丢弃重要的分组流量。

分组处理器体系结构在以下方面尤其新颖，即它允许分组首先被加载到 GPM 104 中，被分发器 106 分配给线程 402，然后在线程开始对新分组进行处理的同时使搜集机构 122 自治地组装分组以便在 BQS 124 中排队。然后 BQS 124 与反馈路径 132 相结合提供允许分组被再循环回 GPM 104 以便由分发器 106 进行线程重新分配的独特特征。

图 9B 概括地描述了分组如何被指定进行再循环。BQS 124（图 9A）包括可能与不同的硬件或软件操作相关联的不同队列。例如，需要被再循环以便进行额外的主处理 458 的多播分组可被加载到高优先级多播队列 430A 或低优先级多播队列 430B 中。需要被再循环回 GPM 104 的控制分配可被加载到控制队列 430C 中，而需要重组的分组可被加载到重组队列 430D 中。根 432 指示 BQS 124 将队列 430 中的分组再循环回 GPM 104。在一个示例中，根被定义为队列的积聚。

其他队列 434 和 436 可能与不同硬件端口的服务成本（QoS）值相关联。队列 434 的根 438 可与第一类输入端口相关联，队列 436 的根 440 可与第二类输入端口相关联。例如，根 438 可与接收自吉比特以太网端口的分组相关联，根 440 可与接收自光学载波（OC）192 端口的分组相关联。根 438 和 440 可以具有一个关联的根 442，根 442 将在初始的到达处理 452 后在队列 434 和 436 中接收到的分组再循环回 GPM 104 以便进行主处理 458。

BQS 124 还可包括其他队列 445，这些队列 445 具有关联的根 446，根 445 不用于再循环而是将关联的分组输出到网络 130（图 1）。其他队列 445 可具有将分组直接再循环回 BQS 124 的关联根 444。

分发器 106 还使用根来分配分组。例如，PPE 110 或线程 402 的不同集合（图 9A）可被分派给不同的根。在到达处理 452 期间或在主处理 458 期间，只要通过将分组分派到 BQS 124 中具有再循环根的特定队列，就可以指定该分组被再循环。同时，也可通过将分组分派到具有被分派给 PPE

110 或线程 402 的特定集合的再循环根的队列，来指定该分组被 PPE 或线程的特定集合所处理。从而，可由 BQS 124 执行的“基于速率”的处理可与分发器 106 执行的基于“处理时间”的调度相混合。

智能分组丢弃

图 10A 示出了传统的分组处理器 469，其中对于分组的所有处理都在单个处理级 472 期间执行。线路 471 上的分组可被存储在到达队列 470 中。分组处理器 469 处理分组，并随后将经处理的分组存储在输出队列 474 中，直到它们被输出到网络上为止。

单个处理级 472 具有可变的处理时间，这种可变的处理时间例如是由不同的分组分段排序或者可能对不同分组执行的不同分组操作引起的。可能出现这样的情形，即分组处理级 472 阻塞，并且导致分组处理器 469 无差别地丢弃分组 476。例如，最新的到达分组可能是最先被丢弃的分组，而不考虑分组优先级或分组控制状态信息。

在图 10B 中，在到达处理 452 中执行初始的速率有限分组处理操作。固定速率到达处理 452 例如可包括访问控制列表或安全性策略。到达处理 452 还可包括检查接收到的分组以找出无效源地址，执行服务质量（QoS）操作，或者确定传入的分组是否包含链路控制数据。在另一个示例中，特定的客户可能未被授权以高于特定的分组速率的速率发送数据。到达处理 452 可识别超过先前协定的传输限度的传入分组流量。

所有这些到达处理操作 452 都要求相对较短的、有限的处理时间。这允许了网络处理器 100 以相对较迅速的有保证的分组处理速率完成到达处理 452 中的任务。从而，在初始的到达处理 452 期间分组不太可能被丢弃。

如果必要，在分组被加载到第 1 级排队 456 中之前，到达处理 452 可丢弃任何不合格的分组 482。这减小了 BQS 124 中的分组负载，并且还减小了主处理 458 期间线程上的负载。

当分组确实需要被丢弃时，到达处理 452 提供了额外的优点，即能够实现更智能的分组丢弃。例如，有时候分组可能仍会在 BQS 124 中溢出。由于在到达处理 452 期间已识别分组的更多特性，因此可以作出更智能的

分组丢弃判决。例如，到达处理 452 可识别存储在 BQS 124 中并被再循环到 GPM 104 以进行主处理 458 的分组的 QoS 信息。如果必要，分组丢弃判决 483 可以基于在到达处理 452 期间得出的 QoS 信息。

负载控制

图 11A 和 11B 示出了再循环路径 132（图 1）被用于负载控制的另一示例。图 11A 中的主处理 458 可能需要具有较长的可变处理时间的任务。例如，多播操作可能要求将同一分组复制多次。多播操作所涉及的线程可能会被占用很长的一段时间。如果大多数或所有线程都被占用来处理多播分组，那就可能没有足够的线程来处理其他主处理操作，并且可能阻碍对原先需要的资源的分配。

如图 11A 所示，主处理 458 要求三个不同的多播操作，每个操作要求多播分组的 16 次复制。图 11B 示出了对于每个主处理会话 458 如何将多播操作限制在 8 次分组复制。如果多播分组必须被复制多于 8 次，则在 8 次复制之后，分组被加载到 BQS 124 中指定用于再循环的特定队列中，如上文在图 9B 中所述。多播分组 460 随后经由再循环路径 132 被再循环回 GPM 104，以再次经过主处理 458。然后在第二次经过主处理 458 期间执行剩余的 8 次分组复制。这允许了线程 402 在更少的时间中完成主处理会话 458，从而释放线程以用于其他分组处理任务。

缓存亲和力（cache affinity）

存在可能促发缓存颠簸（thrashing）的某些分组处理操作。缓存颠簸是指必须反复在缓存和主存储器之间交换数据。这通常是由都必须对同一分组执行的不同的处理操作导致的。向 PPE 110 随机地分发不同的分组处理任务可能增加缓存颠簸。尤其在 PPE 110 是多线程的并且线程共享同一指令缓存时更是如此。例如，线程可能正在运行不同的应用，例如不同的分组协议。由于每种协议可能要求不同的处理步骤，因此在指令缓存中可能发生严重的颠簸。

图 12A 和 12B 示出了可用于减少缓存颠簸的分组处理器的另一个方面。参考图 12A，都优选地执行同一分组处理特征集的线程 402 的群组可被分派到同一个第一主分发群组 524A。例如，主分发群组 524A 中的所有

线程 402 都可被指定用于与一种或多种相同的分组协议相关联的任务。在另一个示例中，分发群组中的线程 402 可被分派给在特定的端口群组上接收的分组。在任何一种情况下，在主分发群组 524A 中指定的线程 402 都与特定的根 520 相关联。次分发群组 524B 也与根 520 相关联，并且可在主分发群组 524A 中的所有线程 402 都不可用时被使用。

类似的主分发群组 526A 和次分发群组 526B 与第二根 522 相关联。当然，主分发群组和次分发群组的数目可取决于根的数目而不同。在一个实施例中，位图（bitmap）被用于识别主分发群组和次分发群组中的线程。

参考图 12B，在块 500 中，设置分发群组的位图。在块 502 中，分发器 106（图 12A）识别与分组相关联的根的主分发群组。在块 504 中，分发器 106 查找主分发群组中的空闲线程 402。如果来自主分发群组的线程 402 之一可用，则在块 506 中分发器 106 将处理任务或分组分派给该可用线程 402。这允许了主分发群组中的线程具有对于与根相关联的操作的亲和力。如果在主分发群组中没有空闲线程 402，则在块 512 中，任务被分派给来自次分发群组的线程 402。随后在块 508 中来自主分发群组或者次分发群组的分派的线程 402 处理分组。

在块 510 中，当需要额外的分组处理时，分组数据在块 507 中被再循环回到主处理 458（图 9A）。分发器 106 随后具有同样的机会将分组分派给来自主分发群组的线程 402 中的另一个。

从而，某些分组处理操作被引导到更可能使用相同的缓存指令的线程子集（亲和力群组）。线程 402 还可动态地重配置分发群组。例如，线程 402 可改变分发群组的映射，使得分发器 106 可向特定的根分派不同的线程。

一个线程可能只被分派到一个或几个主分发群组，但是可能被分派到多得多的次分发群组。这允许了每个根有一个主线程集来执行任务。如果主分发群组中不是所有线程都正被使用，则主分发群组中的可用线程 402 可通过其关联的次分发群组被动态地重分派给其他根。

多层（multi-level）存储器映射

图 13 是多层存储器映射特征的逻辑表示。第一存储器映射层（level）

或级（stage）由转化后备缓冲器（TLB）600 构成，该 TLB 600 将线程 402 所生成的虚拟地址映射到不同的资源。第二存储器映射层或级与识别出的资源相关联，并且使用线程 ID 来将第一存储器映射级生成的地址偏移量映射到识别出的资源中的特定物理地址。在一个示例中，GPM 104 资源中的第二存储器映射级是先前在图 5 中示出的分组句柄数据结构 304。

在一个实施例中，为 PPE 110 中的 CPU 601 所操作的每个线程 402 提供单独的 TLB 600。TLB 600 通过将关联的线程 402 生成的虚拟地址 X 转换成访问分组处理器 100 中的特定资源的偏移量值 X'，来充当第一存储器映射级。TLB 600 被示为位于 PPE 110 中，但是也可利用分组处理器 100 中的其他功能单元来操作。

TLB 600 中的不同条目与不同的分组处理器资源相关联。例如，控制状态寄存器（CSR）条目 602 被线程 402 用来访问存储不同的控制和寄存器数据的存储器资源。通用存储器条目 604 用于访问外部存储器 115（见图 14），例如外部动态随机访问存储器（DRAM）。堆栈条目 606 用于访问外部存储器 115 中的位置，在一个示例中这些位置存储私有堆栈数据。分组数据条目 608 用于访问存储在 GPM 104 中的分组数据。

如上文在图 5 中所述，分组句柄数据结构 300 将 TLB 600 生成的偏移量值 X'映射到分组存储器 311 中与线程 ID 相关联的相应物理存储器空间 610。这允许了每个线程 402 引用同一组虚拟分组地址 0-N，但却能访问分组存储器 311 中的不同物理地址空间 610。

例如，图 13 示出了操作线程 402A 和 402B 的 PPE 110A 和操作线程 402C 的 PPE 110B。每个线程 402A-402C 分别具有关联的 TLB 600A-600C。当然，在分组处理器 100 中可驻留任何数目的 PPE 110，并且 PPE 110 可操作任何数目的线程 402。

每个线程 402 包括与相应的 TLB 600 相关联的线程标识符（线程 ID）。例如，第一线程 402A 生成虚拟地址 X 和关联的线程 ID 值。线程 ID 值将虚拟地址 X 与 TLB 600A 关联起来。TLB 600A 中的条目 608 将虚拟地址 X 映射到访问 GPM 104（资源）的地址偏移量值 X'。线程 402A 的线程 ID 被用于访问 GPM 104 中的关联分组句柄数据结构 304A。分组句柄

数据结构 304A 随后进行第二映射，将地址偏移量 X'映射到分组存储器 311 中的特定物理地址区域 610A。如图 5 所示，分组地址区域 610A 可包括多个非毗邻的数据段，这些数据段包含分派给线程 402A 的分组的分组数据。

类似地，线程 402B 与 TLB 600B 相关联。TLB 600B 中的 TLB 条目 608 可将线程 402B 生成的同一虚拟地址 X 映射到访问同一 GPM 资源 104 的同一偏移量值 X'。但是，线程 402B 的线程 ID 值映射到第二分组句柄数据结构 304B。分组句柄数据结构 304B 随后进行第二映射，将偏移量值 X'映射到分组存储器 311 中的第二物理地址区域 610B。从而，TLB 600 与分组句柄数据结构 304 相结合，将线程 402 生成的虚拟地址映射到 GPM 104 中的唯一物理存储器位置。

图 14 示出了使用双层存储器映射的另一资源。在该实施例中，TLB 600A 被线程 402A 和线程 402B 两者共享。线程 402A 或线程 402B 生成的虚拟地址 Y 被堆栈区域 606 映射到与外部存储器资源 115 相关联的偏移量 Y'。从图 13 中注意到，与分组数据条目 608 相对应的虚拟地址 X 被映射到 GPM 资源 104。外部存储器 115 包括第二存储器映射 620，该映射将偏移量值 Y'映射到存储器 624 中的不同物理地址位置 622。例如，线程 402A 的偏移量 Y'被映射到物理地址区域 622A，而线程 402B 的同一偏移量 Y'被映射到物理地址区域 622B。

从而，第一存储器映射级将虚拟地址映射到不同的资源，并且可能将其映射到特定资源内的不同位置，然后第二存储器映射级将由第一级生成的偏移量值映射到与线程 ID 相关联的特定物理存储器位置。

协同处理（co-processing）

图 15 示出了根据线程 402 发布的命令独立地访问 GPM 104 的一个或多个协同处理器资源 634。这允许了混合线程处理和协同处理器处理。在一个实施例中，GPM 104 和协同处理器 634 位于同一芯片上。这允许了协同处理器 634 被直接附接到 GPM 104，而无需任何额外的缓冲。

协同处理器 634 充当并被线程 402 视为任何其他资源。如上所述，资源是任何接收来自线程 402 的命令并发回结果的元件。例如，外部存储器

115（图 1）是一个资源，GPM 104 是另一个资源。充当资源的协同处理器 634、自治地访问 GPM 104 中的分组数据的协同处理器 634 和 PPE 110 的多线程能力的组合提供了改进的协同处理器操作性。

例如，第一处理级中的第一线程 402A 开始处理分组 630。如上文在图 13 中所述，线程 402A 的线程 ID 在 GPM 104 中被用于标识分组 630。在第二处理级中，线程 402A 向协同处理器 634 发送包括线程 ID 和到分组 630 中的偏移量的命令 636。协同处理器 634 包含与协同处理器命令 636 相对应的算法。协同处理器 634 利用线程 402A 提供的线程 ID 和偏移量来访问并随后开始自治地处理分组 630。

取决于软件和分组操作，线程 402A 可继续与协同处理器 634 并行地处理分组 630。在协同处理器 634 处理分组 630 的同时，线程 402A 也可启动处理相同或不同分组的其他协同处理器 634。

协同处理器 634 在其完成对分组 630 的处理时向线程 402A 发回通知 638。然后第三处理级中的线程 402A 完成分组 630 所需的任何额外的处理。

PPE 110 中的多线程化与协同处理器 634 的自治操作相结合，还允许了其他线程在一个线程可能停止以等待协同处理器 634 返回结果的同时运行。例如，第二线程 402B 可继续或开始处理另一分组 632，即使线程 402A 被停止以等待协同处理器 634 完成对分组 630 的处理。

协同处理器 634 执行的处理的一个示例可包括密码术（密码）操作。通常密码处理器位于传统的分组处理器之前。关于这类体系结构存在若干问题。例如，在密码操作之前可能要求分组优先级区分。在密码操作之前可能还要求其他预过滤或解封装操作。但是，在密码操作之后，就在分组被发送出分组处理器前夕，可能要求其他处理操作，例如多链路点到点协议（MLPPP）。

如上文图 15 中所示，协同处理器 634 进行的密码处理可以很容易地在由 PPE 110 执行的其他分组处理之前或之后被混合。从而，分组处理器体系结构在将协同处理操作与其他通用 PPE 分组处理操作相集成方面更加有效。

在图 15 中，PPE 110 通过资源互连 108 访问协同处理器 634 和 GPM 104。在另一个实施例中，资源互连 108 可将分组处理元件 110 与 GPM 104 相耦合，并且并行地将协同处理器 634 与 GPM 104 相耦合。

缓存硬件辅助装置

一般来说，网络处理器不是缓存一致的。如果多处理器系统工作在其中每个处理器尝试完全处理一个分组然后才进行到另一分组的运行-完成（run to completion）模型上，那么这就可能造成问题。某些多处理器体系结构可以处理分组的一段，然后将分组移交给另一处理器。当多个不同的处理器各自对同一分组进行一段工作时，不会有那么多一致性问题。例如，当每个处理器对每个分组进行同一工作时，在同一处理器内一般可维持相同的处理器状态，例如数据结构的状态。

在分组处理器 100 的一个实施例中，一个线程可对分组执行所有或大部分工作。如果数据结构延伸到跨分组，则数据需要被高效地定位回主存储器中以供其他线程访问。

图 16 示出了转化后备缓冲器（TLB）664，该 TLB 664 包括被中央处理单元（CPU）652 访问的 TLB 条目 665。TLB 条目 665 包括虚拟地址范围 666、相应的物理地址偏移量 668 以及可能的额外控制部分 669。控制部分 669 可被用于确定物理地址是否是可读的、可写的、分配和存储器排序参数。存储器访问可在 TLB 中被指定为被缓冲的（不严格顺序）或不被缓冲的（写同步严格排序）。

TLB 664 将虚拟地址 666 映射到与不同资源相关联的物理地址偏移量 668。例如，如上文图 13 中所述，不同的 TLB 条目 665 可与诸如 GPM 104 和外部存储器 115 之类的不同资源相关联。其他 TLB 条目可与资源之一内的特定区域相关联，例如与堆栈区域或 CSR 区域相关联。

缓存 650 包括标签 660，该标签标识包含在缓存行（cache line）658 中的数据的物理地址的最高位。在一个示例中，缓存 650 访问外部存储器 115 或片上 GPM 104。缓存行 658 可用于指令数据或分组数据。

缓存控制器 654 是由缓存命令 662 激活的硬件状态机。缓存控制器 654 代表在逻辑上可由硬件和/或软件执行的操作。例如，一种指令体系结

构可具有软件控制的 TLB 664。缓存控制器 654 可处理正常数据缓存操作（行分配、收回和存储器排序）以及下文描述的页变址（page-indexed）操作。

CPU 652 发送引用 TLB 条目 665 之一的单个缓存命令 662。每个 TLB 条目 665 可具有在缓存命令 662 中标识的关联号码。根据缓存命令 662，控制器 654 对与 TLB 条目 665 相关联的所有缓存行 658 执行缓存操作。

参考图 17，在块 670 中，控制器 654 接收来自 CPU 652 的标识特定的 TLB 条目 665 的缓存命令 662。在块 672 中，控制器 654 通过将某个变量值设置为 0（cacheline=0）来初始化它自身。这实际上将控制器 654 指向了缓存 650 中的第一缓存行 658。在块 674 中，控制器 654 为 cacheline=0 读取标签 660 中的物理地址。

在块 676 中，控制器 654 将标签 660 中的物理地址与缓存命令 662 中指定的 TLB 条目的物理地址 668 相比较。在一个实施例中，缓存行的物理地址是利用多周期（multi-cycle）操作获得的。此外，如果缓存是集合关联型的（set-associative），则缓存行变量被划分成具有若干个路选择（way selection）位和若干个集合选择（set selection）位，以将 2-D 缓存行阵列映射到整数计数器上。

如果在块 676 中存在匹配，并且如果缓存行是脏的（dirty），则在块 678 中控制器执行缓存命令 662 中指定的缓存操作。如果不存在匹配或者如果缓存行不是脏的，则在块 680 中控制器 654 递增变量（cacheline = cacheline+1）。

然后，与下一缓存行 658 相关联的物理地址被与指定 TLB 条目 665 的物理地址 668 相比较。此过程在块 682 中重复，直到对与指定 TLB 条目 665 相对应的所有缓存行 658 都执行了缓存命令 662 中的缓存操作。从而，控制器 654 自动检查所有标签 660 并根据单个 CPU 命令 662 对与指定 TLB 条目相对应的所有缓存行执行缓存操作。

控制器 654 可执行与不同缓存命令 662 相对应的不同类型的缓存操作。例如，缓存命令 662 可以是清除（flush）命令，该命令使得在缓存行 658 中的内容保持有效的同时控制器 654 将缓存行 658 的内容发回主存储

器。缓存命令 662 还可以是清除并无效命令，该命令使得控制器 654 将缓存行 658 的内容存储回主存储器中，然后使这些缓存行无效。

TLB 缓存控制器 654 确保了缓存 650 中包含的共享数据结构等被正确地存储回存储器 115 或 104 中并可供其他线程使用。这允许了软件在非一致条件下更高效地工作。控制器 654 还使得 PPE 软件可免于管理每个缓存操作。

变化

可以实现上述分组处理器 100 的许多变体。例如，分组可被映射到 PPE 110 的虚拟或物理地址空间中。资源互连 108 可采取若干种形式中的任何一种。缓存体系结构可采取若干种形式中的任何一种。PPE 110 可实现缓存一致性，也可不实现。GPM 104 可按多种方式实现。PPE 110 可具有不同数目的线程，并且它们的内部体系结构可能有所不同。BQS 124 可被集成到包含其他处理元件的同一芯片中也可不被集成，并且可以更多地以硬件实现，也可按更软件密集的方式实现。

上述系统可使用执行某些或全部操作的专用处理器系统、微控制器、可编程逻辑器件或者微处理器。上述操作中的某些可用软件实现，其他操作可用硬件实现。

为了方便，操作被描述成各种互连的功能块或不同的软件模块。但是，这并不是必要的，可能存在这样的情况，即这些功能块或模块被等同地聚集成边界不清的单个逻辑器件、程序或操作。在任何情况下，灵活的接口的功能块和软件模块或特征可由其自身实现，或者可与其他操作相结合利用硬件或软件来实现。

在利用本发明的优选实施例描述和说明了本发明的原理之后，应当很清楚看到，本发明的布置和细节都可被修改，而不会脱离这种原理。发明人要求保护落在所附权利要求的精神和范围内的所有修改和变化。

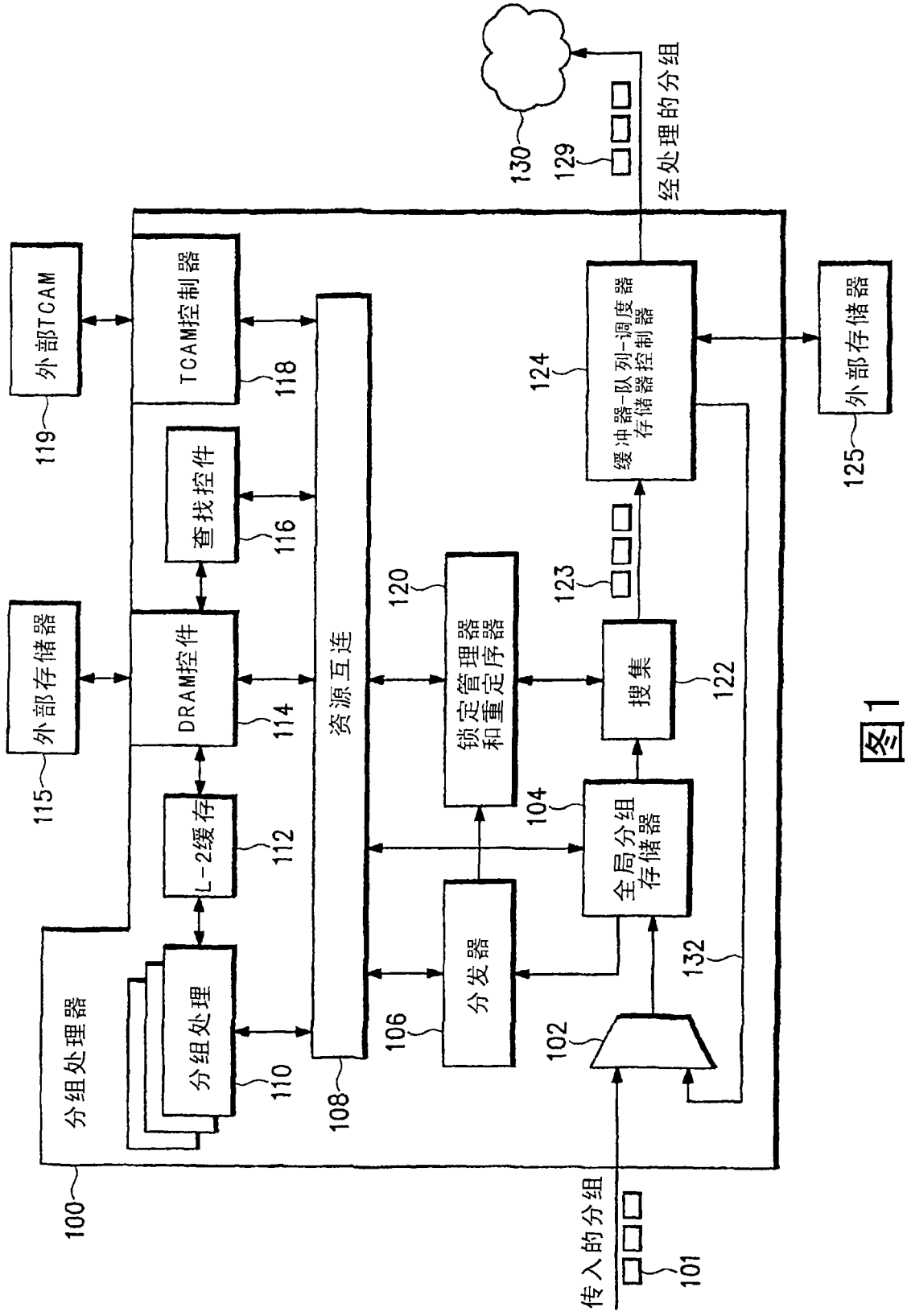

外部TCAM
119
TCAM控制器
118
查找控件
116
外部存储器
115
DRAM控件
114
L-2缓存
112
分组处理器
分组处理
110
100
资源互连
108
锁定管理器
和重定序器
120
分发器
106
全局分组
存储器
104
搜集
122
123
缓冲器-队列-调度器
存储器控制器
124
外部存储器
125
130
129
经处理的分组
传入的分组
101
102
132

图1

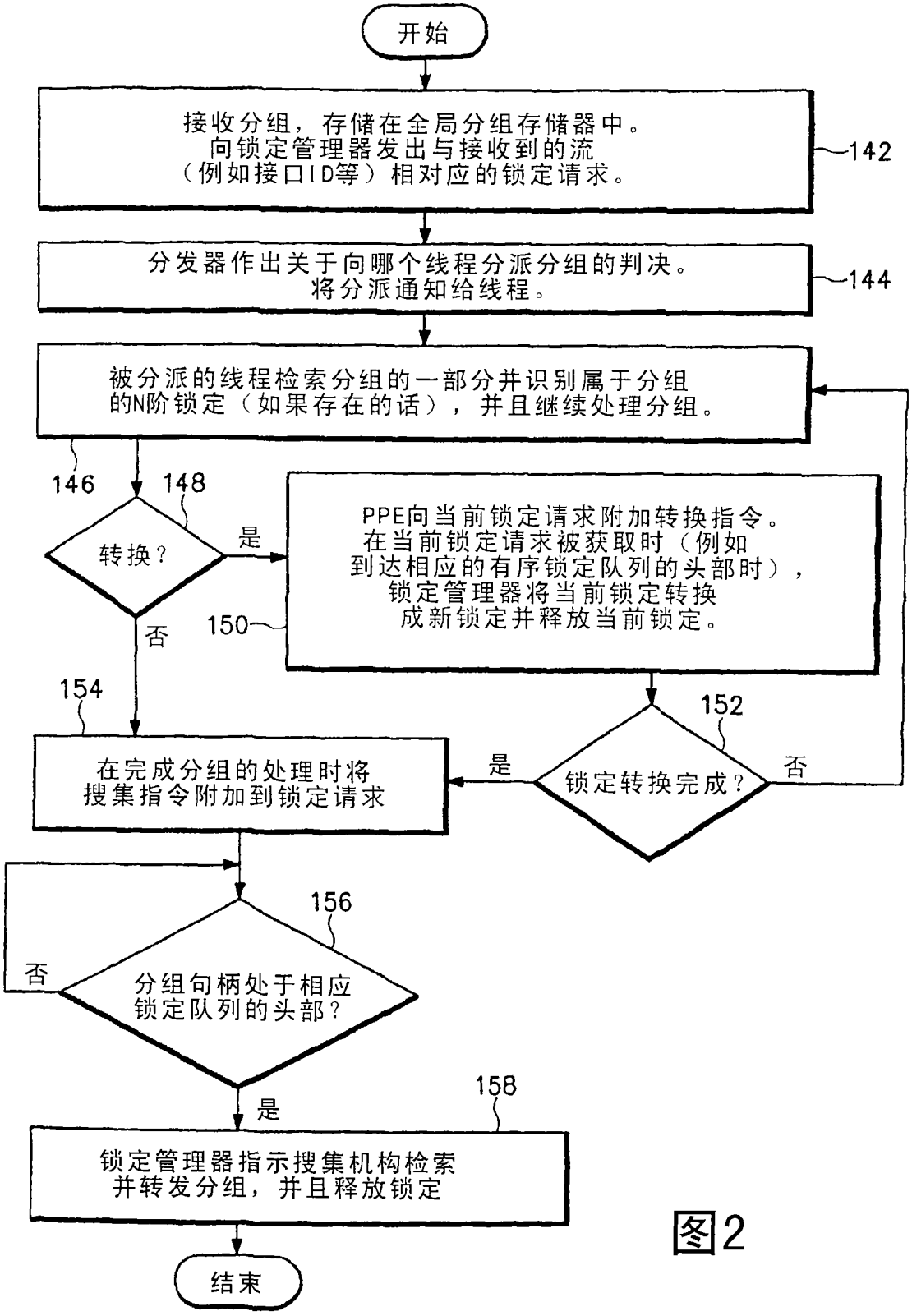

开始
接收分组，存储在全局分组存储器中。
向锁定管理器发出与接收到的流
（例如接口ID等）相对应的锁定请求。
142
分发器作出关于向哪个线程分派分组的判决。
将分派通知给线程。
144
被分派的线程检索分组的一部分并识别属于分组
的N阶锁定（如果存在的话），并且继续处理分组。
146
148
转换？
是
否
150
PPE向当前锁定请求附加转换指令。
在当前锁定请求被获取时（例如
到达相应的有序锁定队列的头部时），
锁定管理器将当前锁定转换
成新锁定并释放当前锁定。
152
锁定转换完成？
是
否
154
在完成分组的处理时将
搜集指令附加到锁定请求
156
分组句柄处于相应
锁定队列的头部？
否
是
158
锁定管理器指示搜集机构检索
并转发分组，并且释放锁定
结束

图2

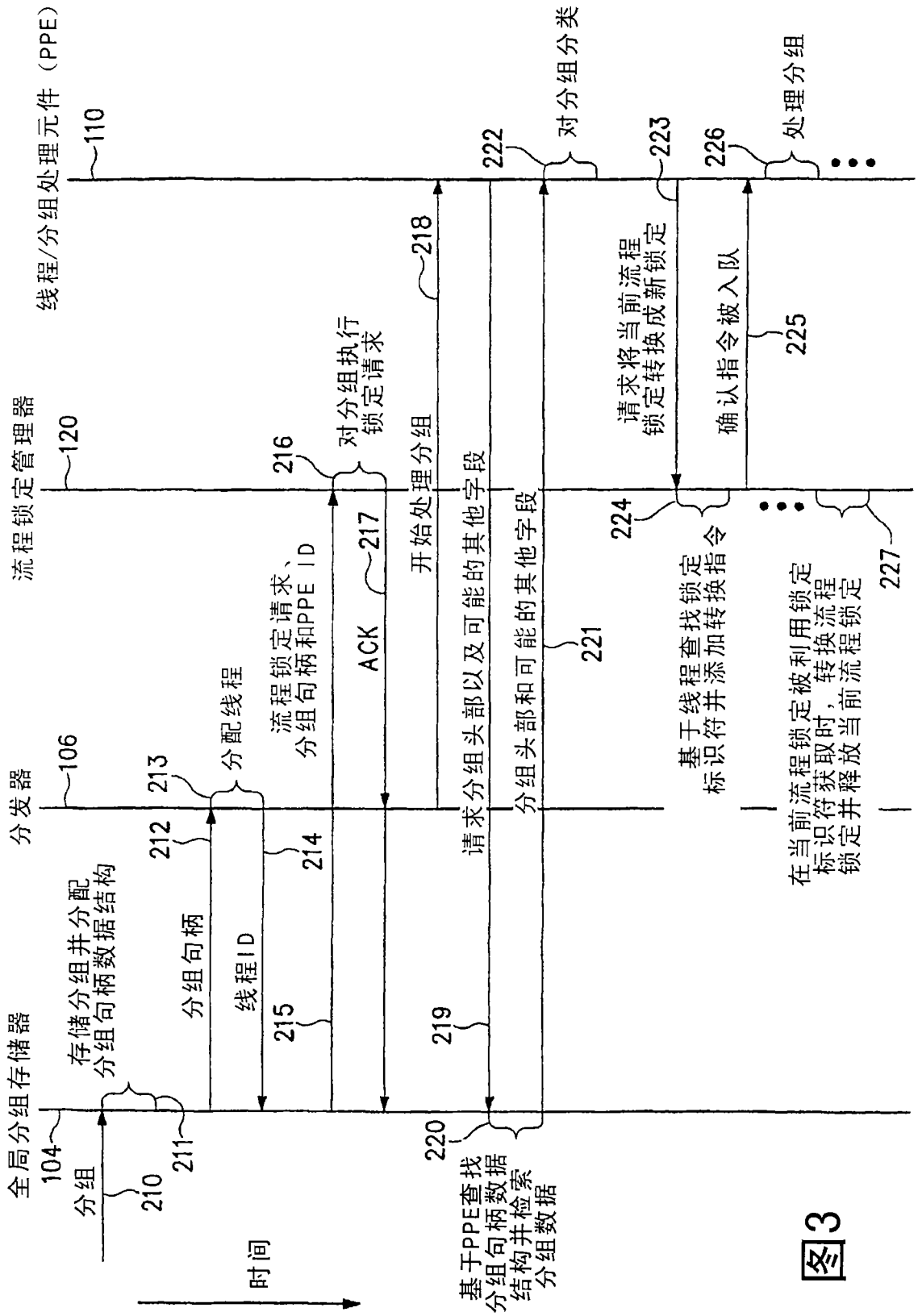

全局分组存储器
104
分组
210
211
存储分组并分配
分组句柄数据结构
分发器
106
212
分组句柄
213
分配线程
线程ID
214
215
流程锁定管理器
120
流程锁定请求、
分组句柄和PPE ID
216
ACK
217
对分组执行
锁定请求
218
开始处理分组
线程/分组处理元件（PPE）
110
222
对分组分类
请求分组头部以及可能的其他字段
219
分组头部和可能的其他字段
221
基于PPE查找
分组句柄数据
结构并检索
分组数据
220
请求将当前流程
锁定转换成新锁定
223
224
基于线程查找锁定
标识符并添加转换指令
确认指令被入队
225
226
处理分组
在当前流程锁定被利用锁定
标识符获取时，转换流程
锁定并释放当前流程锁定
227
时间

图3

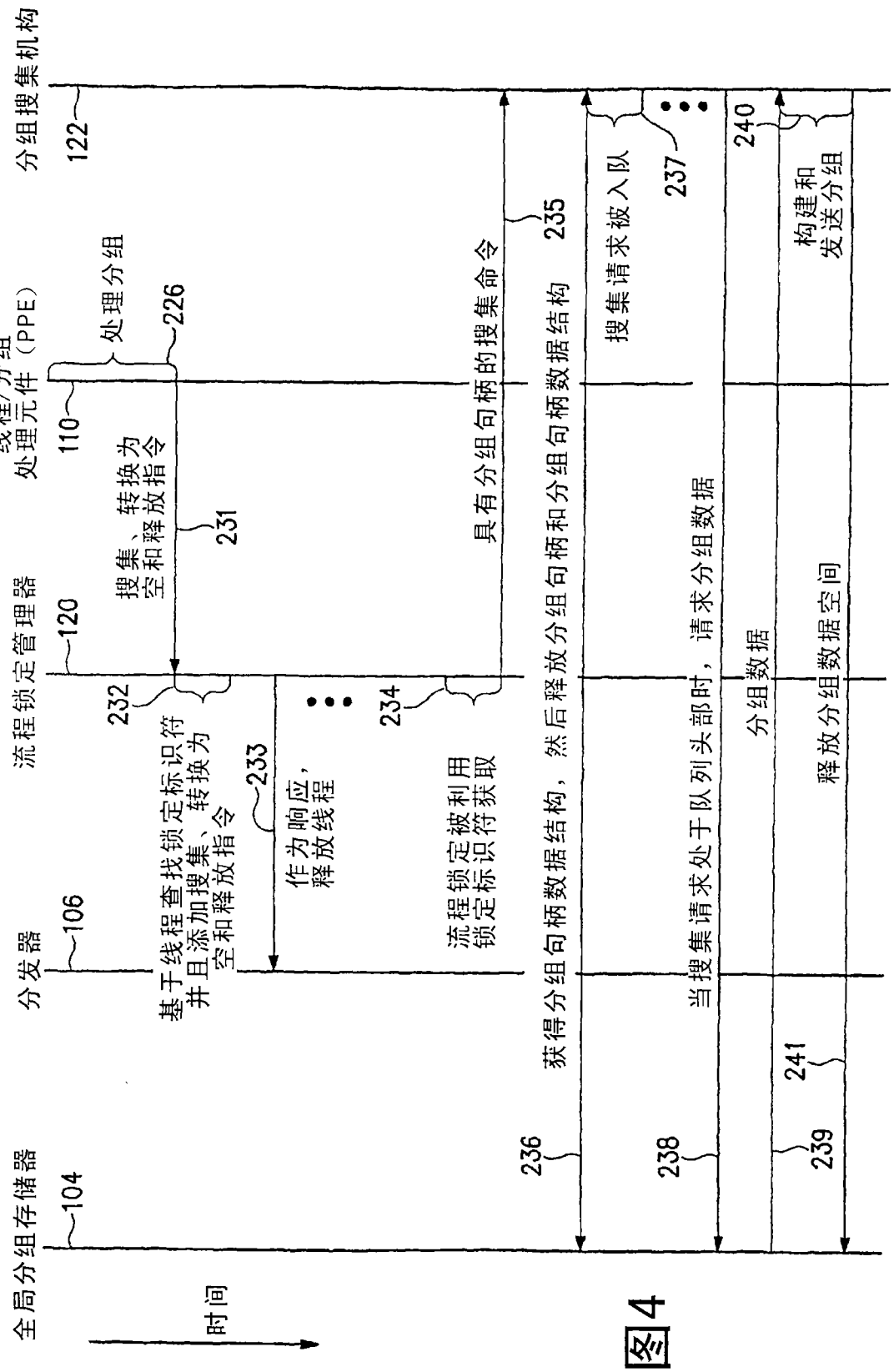

全局分组存储器
104
分发器
106
流程锁定管理器
120
线程/分组处理元件（PPE）
110
分组搜集机构
122
基于线程查找锁定标识符并且添加搜集、转换为空和释放指令
233
作为响应，释放线程
流程锁定被利用锁定标识符获取
234
搜集、转换为空和释放指令
231
232
处理分组
226
具有分组句柄的搜集命令
235
搜集请求被入队
237
获得分组句柄数据结构，然后释放分组句柄和分组句柄数据结构
236
当搜集请求处于队列头部时，请求分组数据
238
分组数据
240
释放分组数据空间
241
239
构建和发送分组
时间

图4

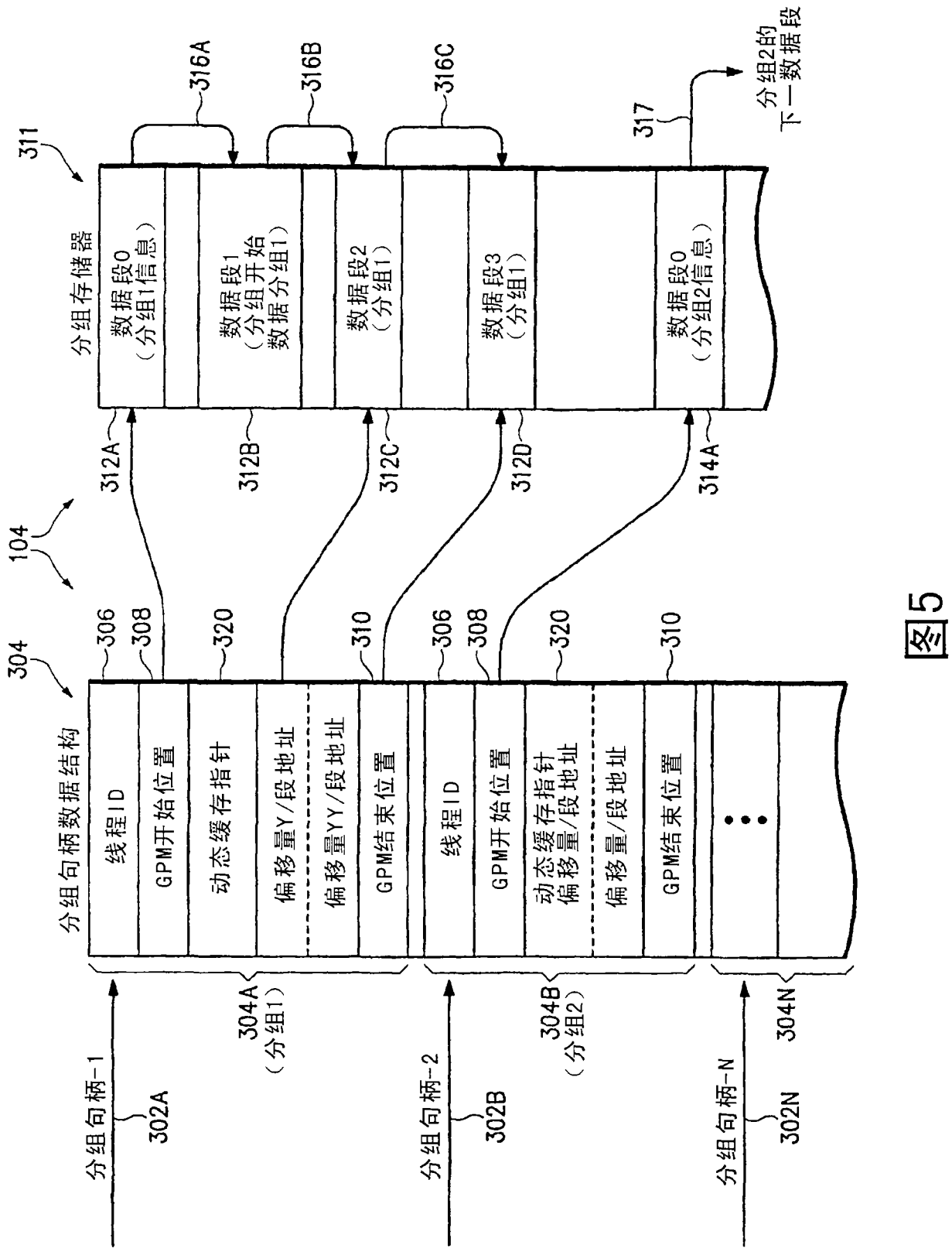

311
分组存储器
316A
316B
316C
317
分组2的
下一数据段
数据段0
（分组1信息）
数据段1
（分组开始
数据分组1）
数据段2
（分组1）
数据段3
（分组1）
数据段0
（分组2信息）
312A
312B
312C
312D
314A
104
304
分组句柄数据结构
306
308
320
310
线程ID
GPM开始位置
动态缓存指针
偏移量Y/段地址
偏移量YY/段地址
GPM结束位置
动态缓存指针
偏移量/段地址
304A
（分组1）
304B
（分组2）
304N
分组句柄-1
302A
分组句柄-2
302B
分组句柄-N
302N

图5

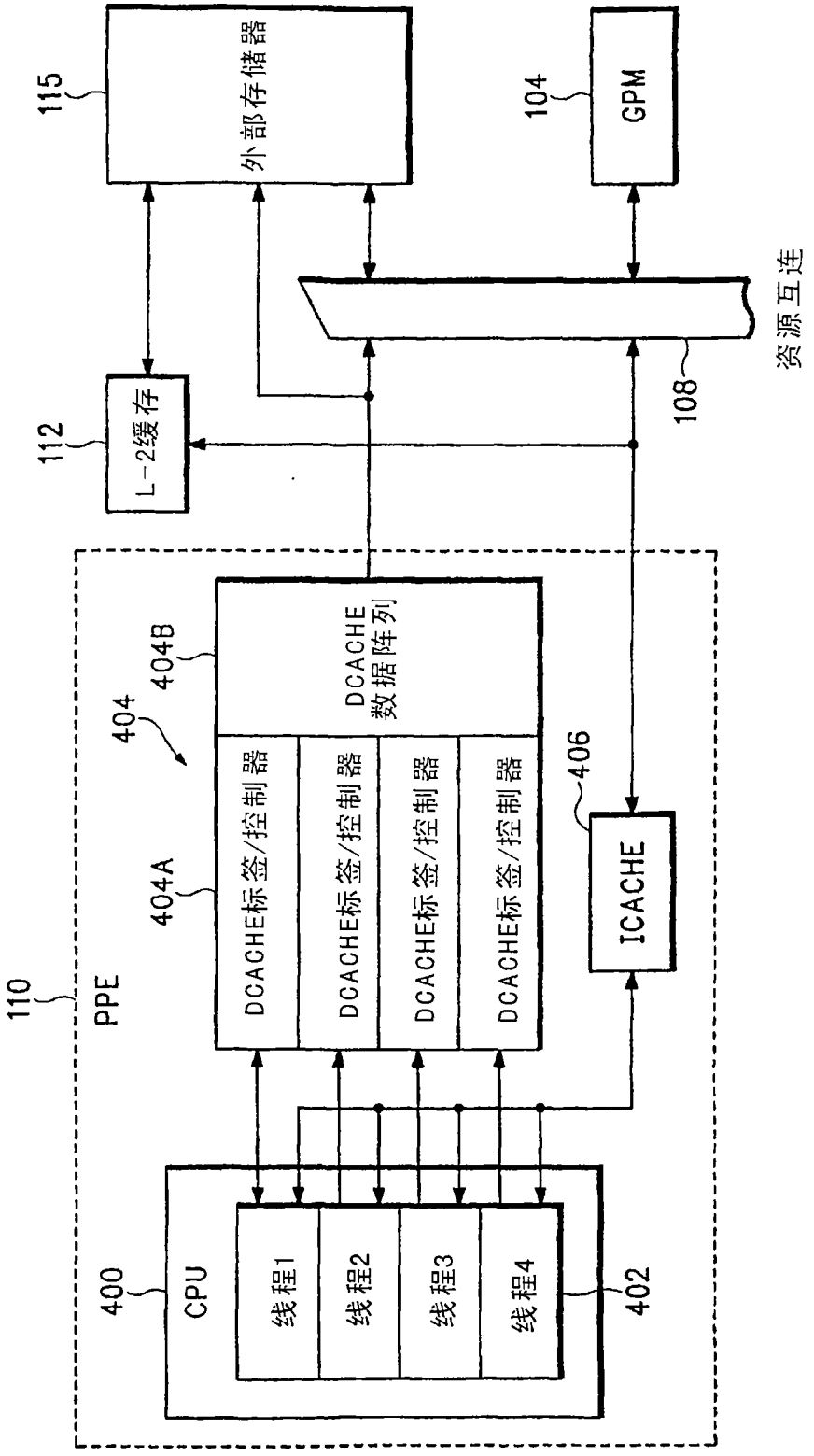

115
外部存储器
104
GPM
资源互连
108
112
L-2缓存
110
PPE
404
404B
DCACHE
数据阵列
404A
DCACHE标签/控制器
DCACHE标签/控制器
DCACHE标签/控制器
DCACHE标签/控制器
406
ICACHE
400
CPU
线程1
线程2
线程3
线程4
402

图6

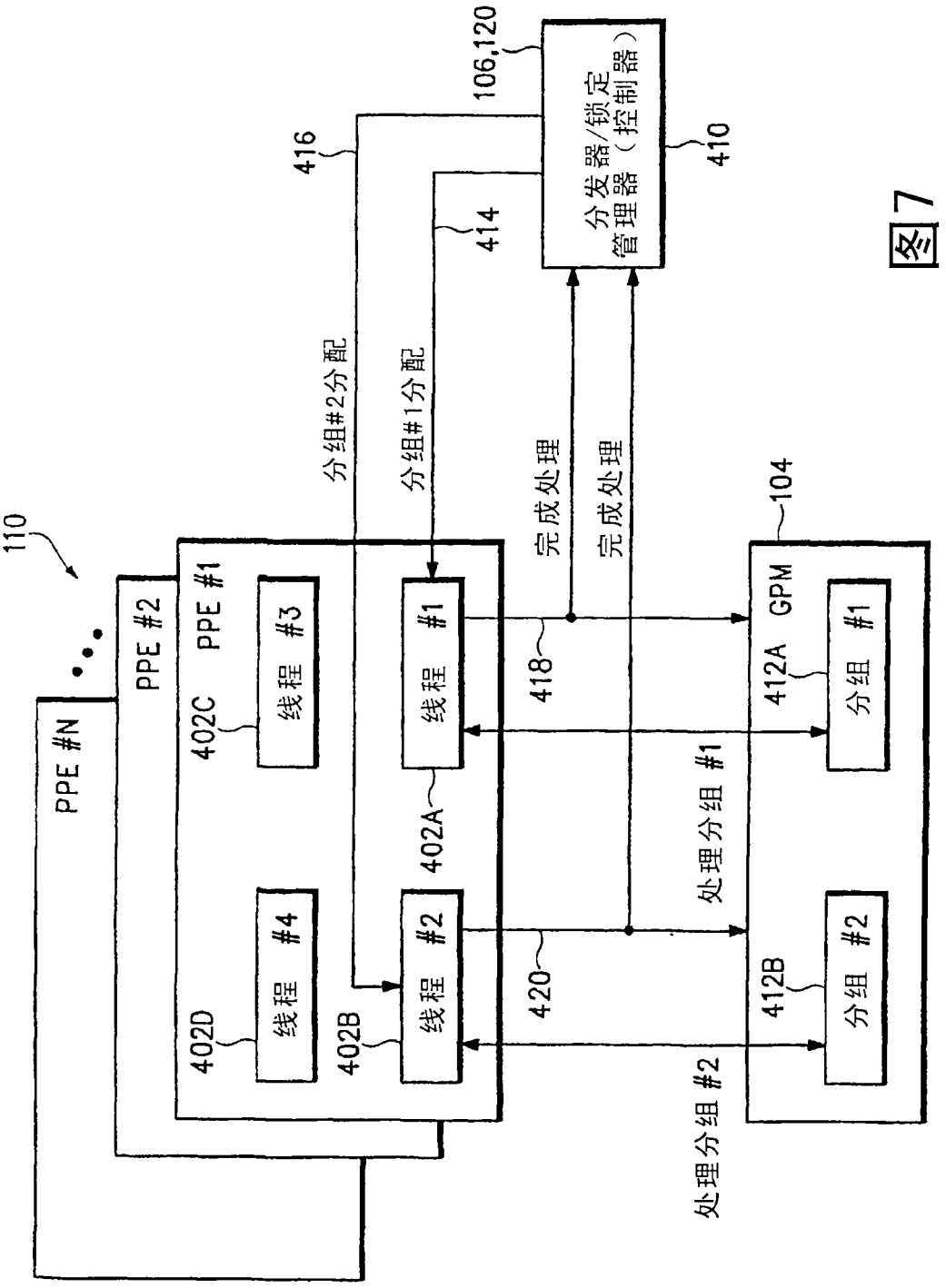

110
PPE #N
PPE #2
PPE #1
402C
线程 #3
402D
线程 #4
402B
线程 #2
402A
线程 #1
416
414
106,120
分发器/锁定
管理器（控制器）
410
分组#2分配
分组#1分配
完成处理
完成处理
418
420
104
412A GPM
分组 #1
412B
分组 #2
处理分组 #1
处理分组 #2

图7

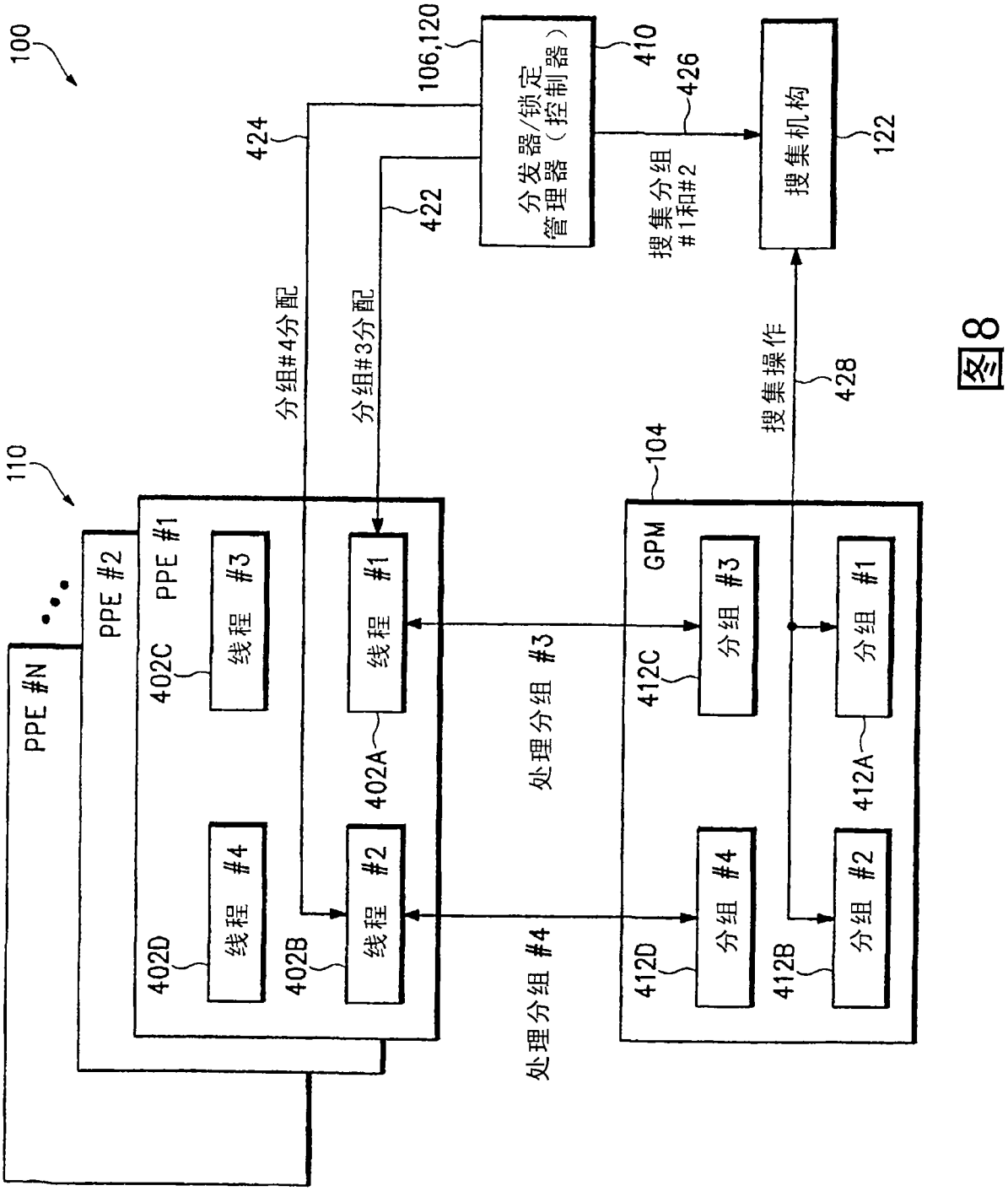

100
106,120
410
分发器/锁定管理器（控制器）
424
422
426
搜集分组#1和#2
搜集机构
122
分组#4分配
分组#3分配
搜集操作
428
110
104
PPE #N
PPE #2
PPE #1
GPM
402C
线程 #3
线程 #1
412C
分组 #3
分组 #1
402A
412A
处理分组 #3
402D
线程 #4
402B
线程 #2
412D
分组 #4
412B
分组 #2
处理分组 #4

图8

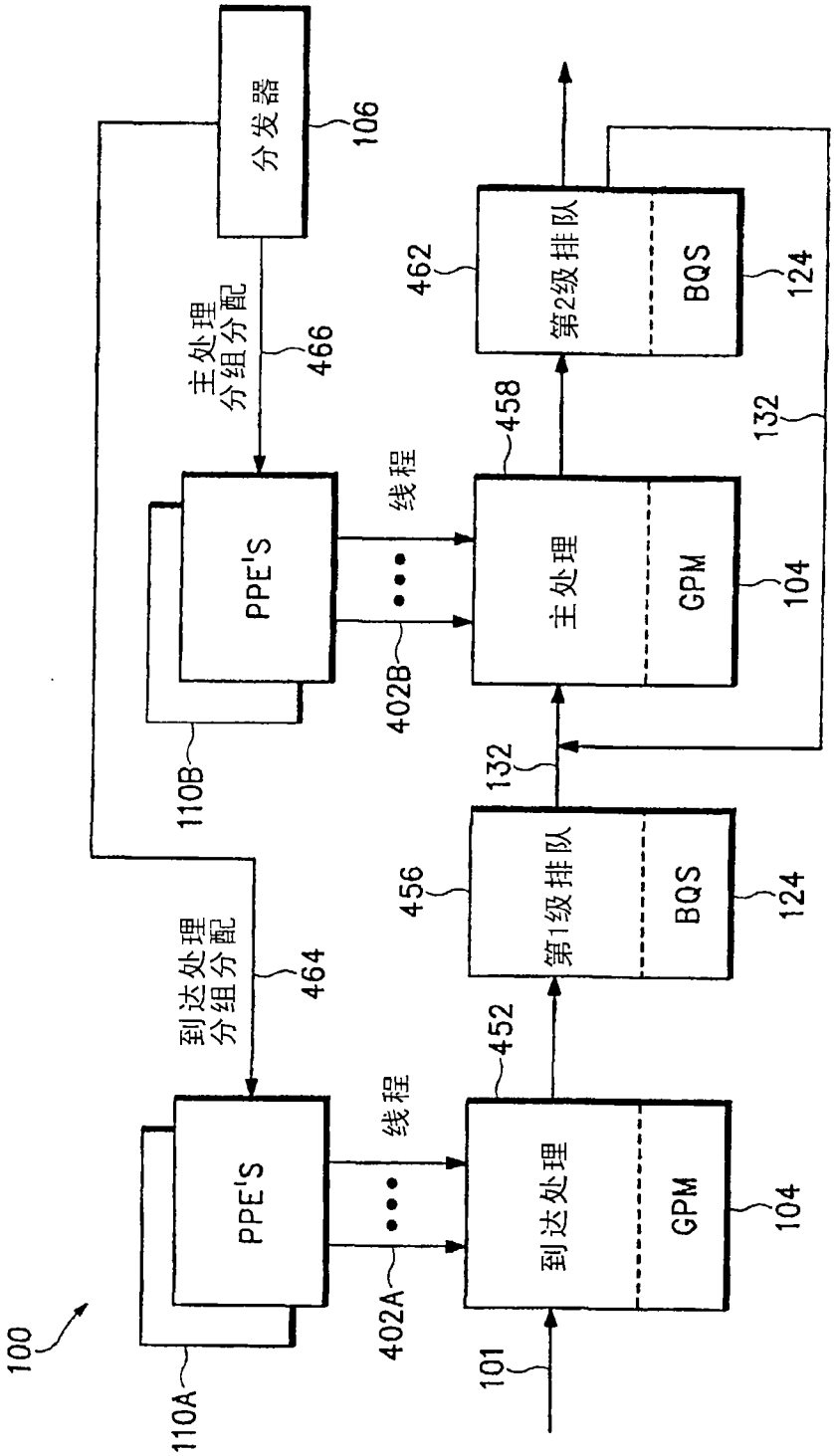

100
101
110A
110B
PPE'S
PPE'S
402A
402B
线程
线程
到达处理
GPM
104
主处理
GPM
104
452
458
第1级排队
BQS
124
456
第2级排队
BQS
124
462
132
132
分发器
106
到达处理
分组分配
464
主处理
分组分配
466

图9A

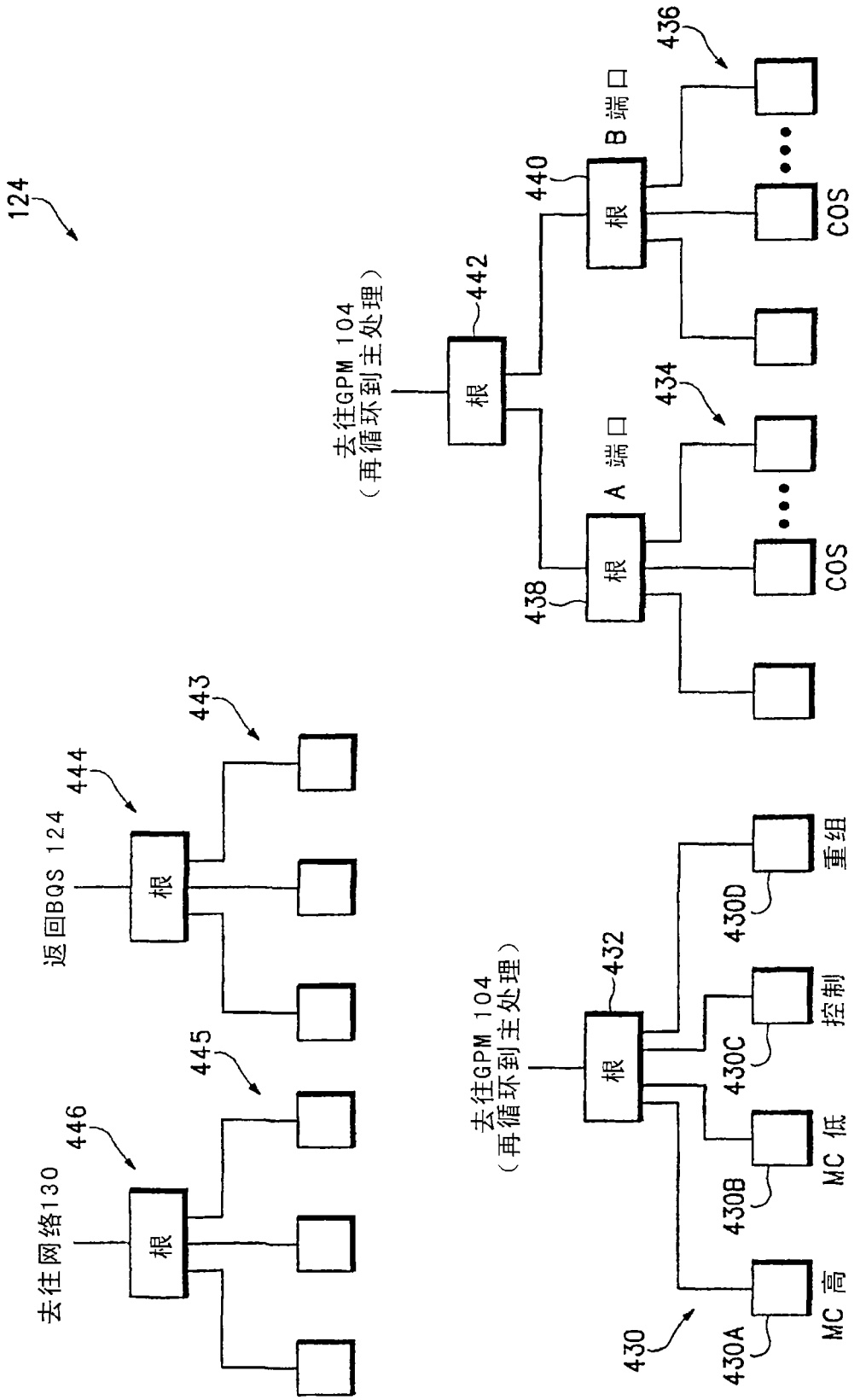

124
去往网络130
返回BQS 124
去往GPM 104
（再循环到主处理）
去往GPM 104
（再循环到主处理）
根
A 端口
B 端口
COS
MC 高
MC 低
控制
重组
430
430A
430B
430C
430D
432
434
436
438
440
442
443
444
445
446

图9B

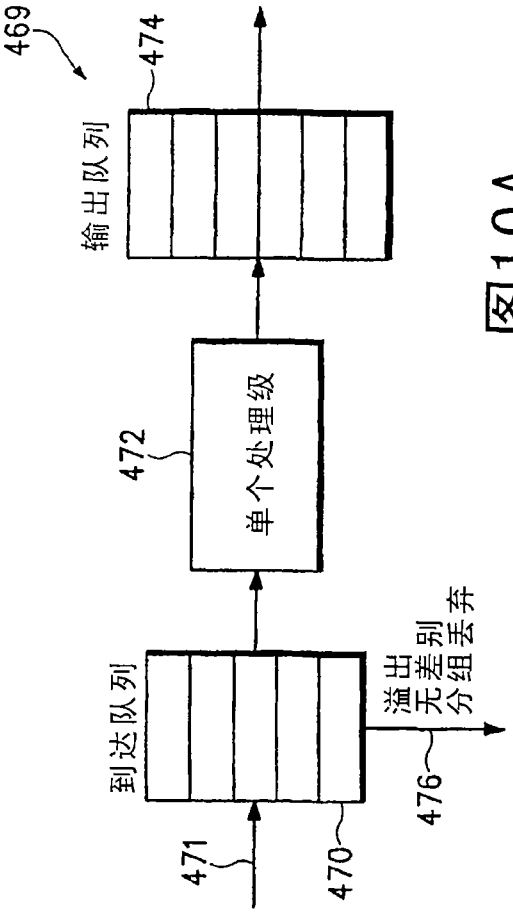

469
474
输出队列
472
单个处理级
到达队列
溢出
无差别
分组丢弃
476
471
470

图10A

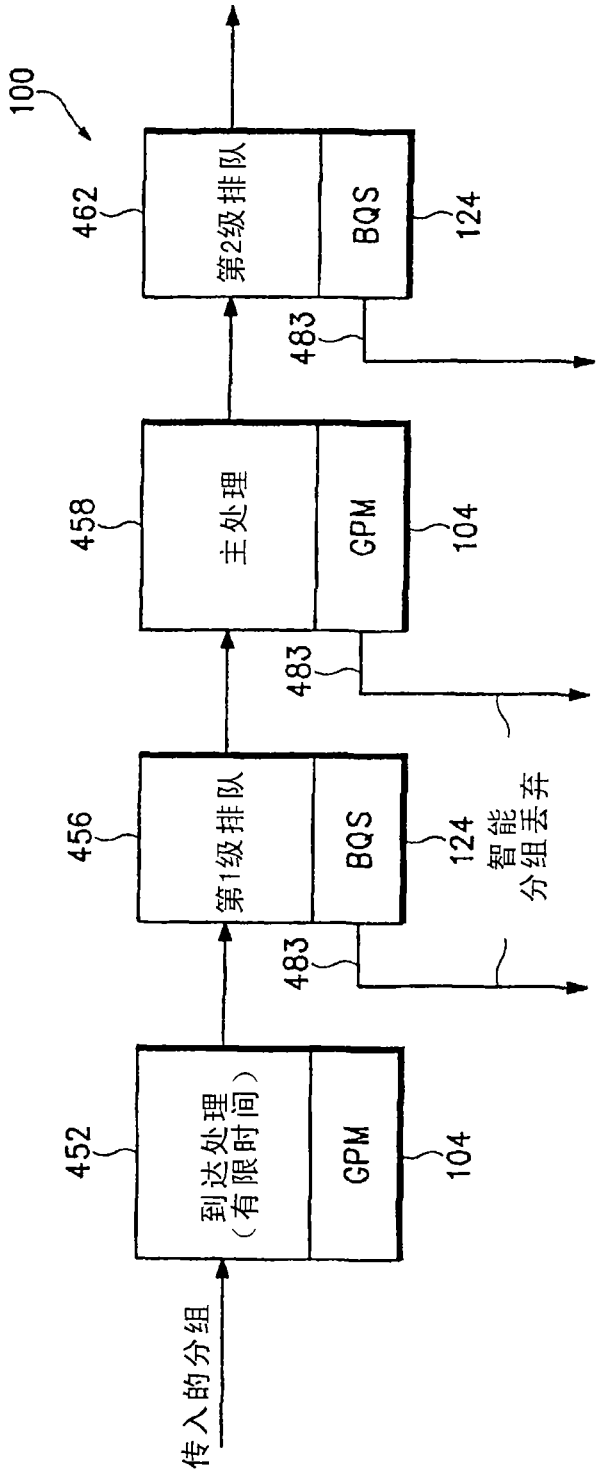

100
462
第2级排队
BQS
124
483
458
主处理
GPM
104
483
456
第1级排队
BQS
124
智能
分组丢弃
483
452
到达处理
（有限时间）
GPM
104
传入的分组

图10B

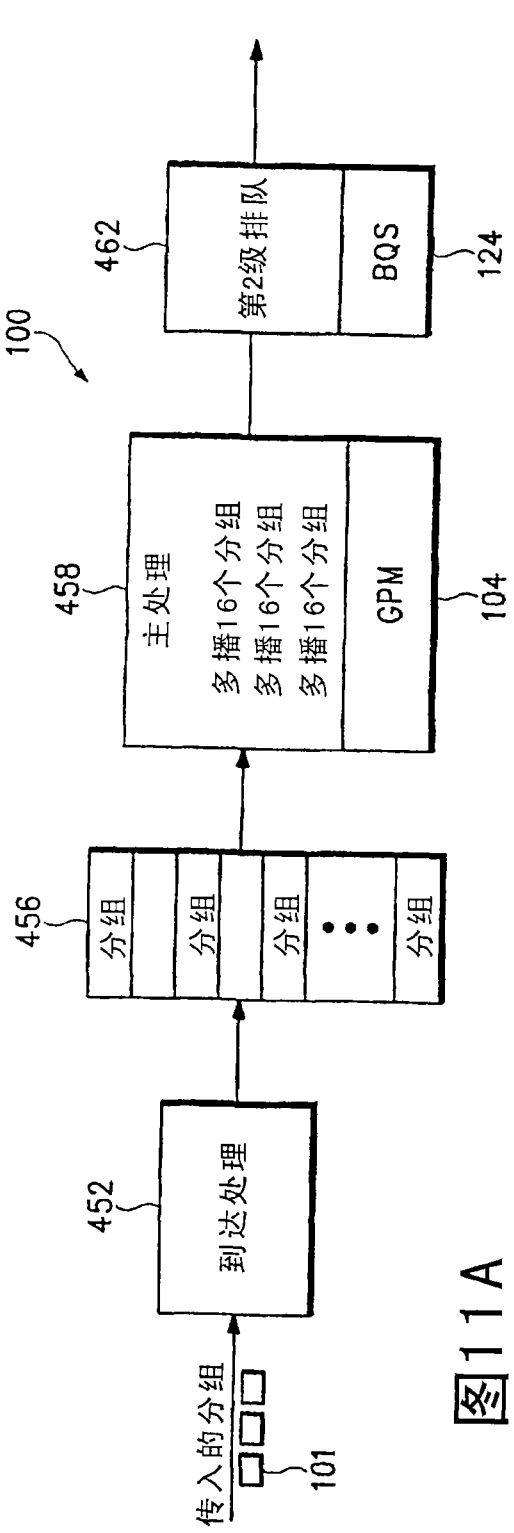

100
传入的分组
101
452
到达处理
456
分组
分组
分组
分组
458
主处理
多播16个分组
多播16个分组
多播16个分组
GPM
104
462
第2级排队
BQS
124

图11A

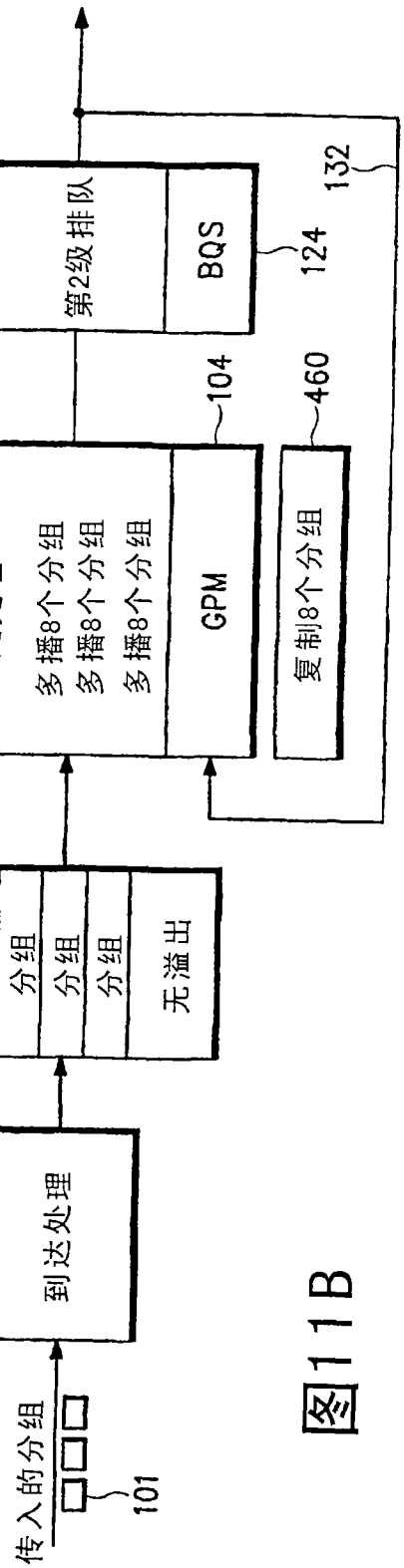

传入的分组
101
452
到达处理
456
分组
分组
分组
分组
无溢出
458
主处理
多播8个分组
多播8个分组
多播8个分组
GPM
104
460
复制8个分组
462
第2级排队
BQS
124
132

图11B

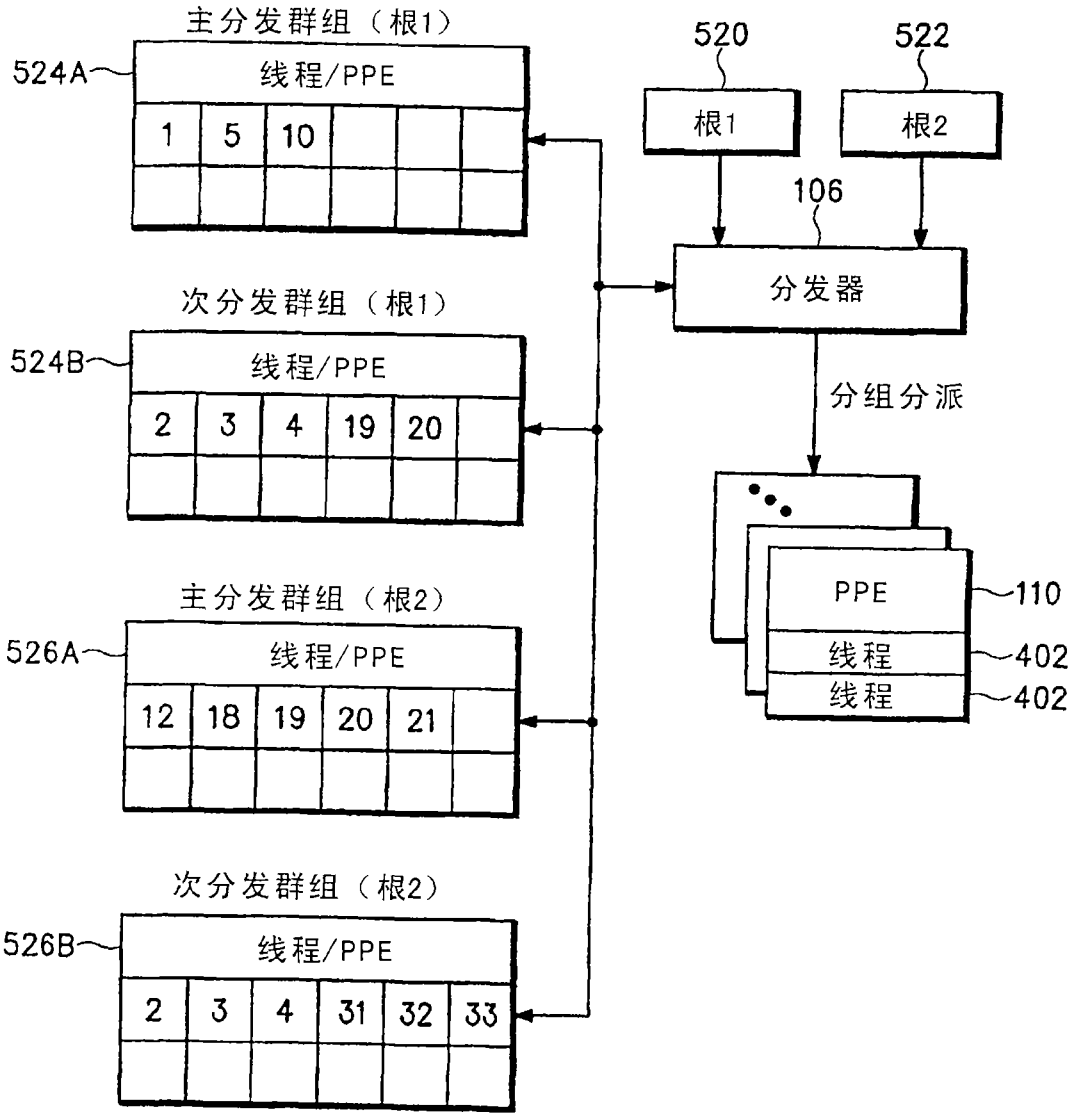

主分发群组（根1）
524A
线程/PPE
1
5
10
次分发群组（根1）
524B
线程/PPE
2
3
4
19
20
主分发群组（根2）
526A
线程/PPE
12
18
19
20
21
次分发群组（根2）
526B
线程/PPE
2
3
4
31
32
33
520
522
根1
根2
106
分发器
分组分派
PPE
110
线程
402
线程
402

图12A

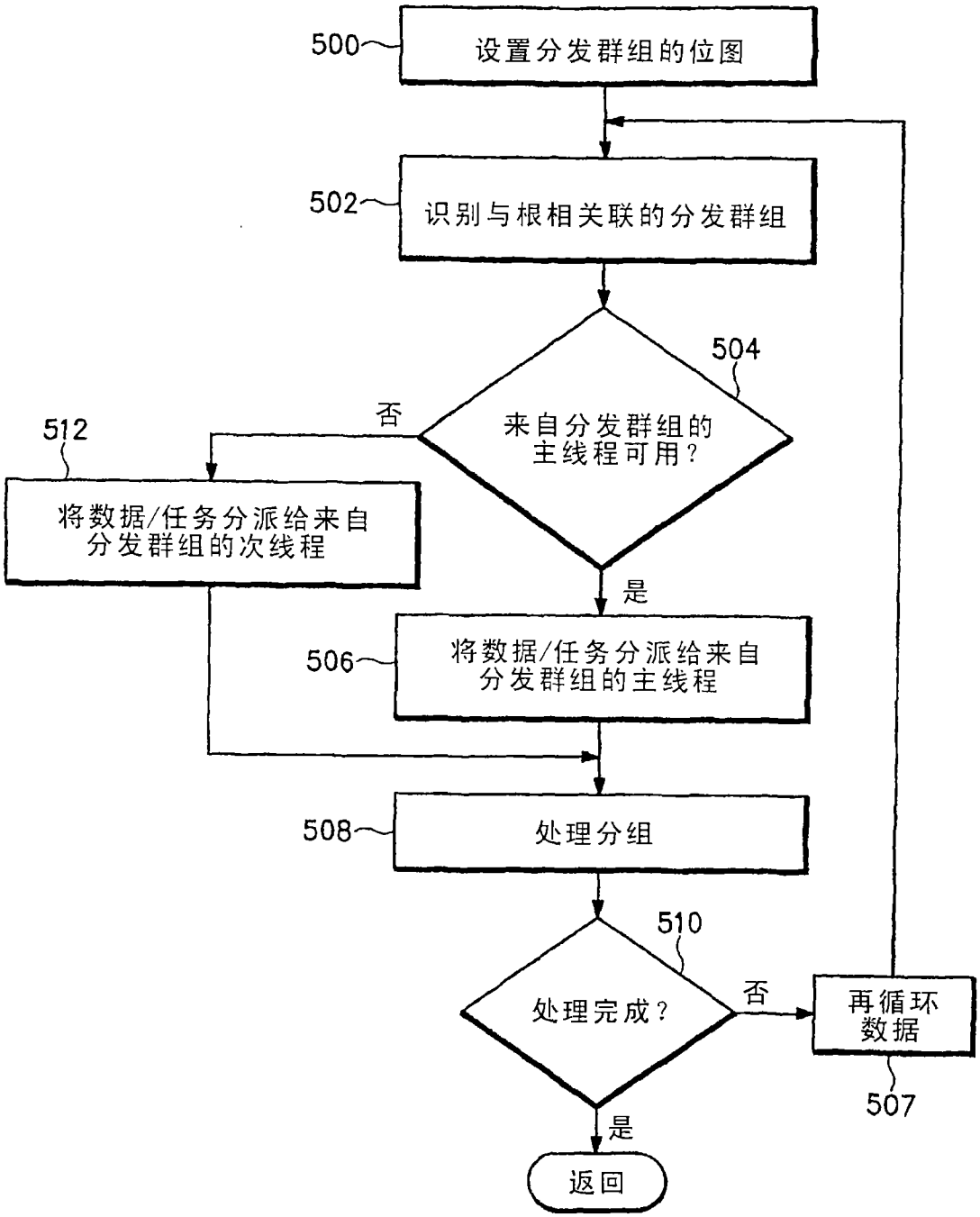

500
设置分发群组的位图
502
识别与根相关联的分发群组
504
来自分发群组的主线程可用？
否
512
将数据/任务分派给来自分发群组的次线程
是
506
将数据/任务分派给来自分发群组的主线程
508
处理分组
510
处理完成？
否
再循环数据
507
是
返回

图12B

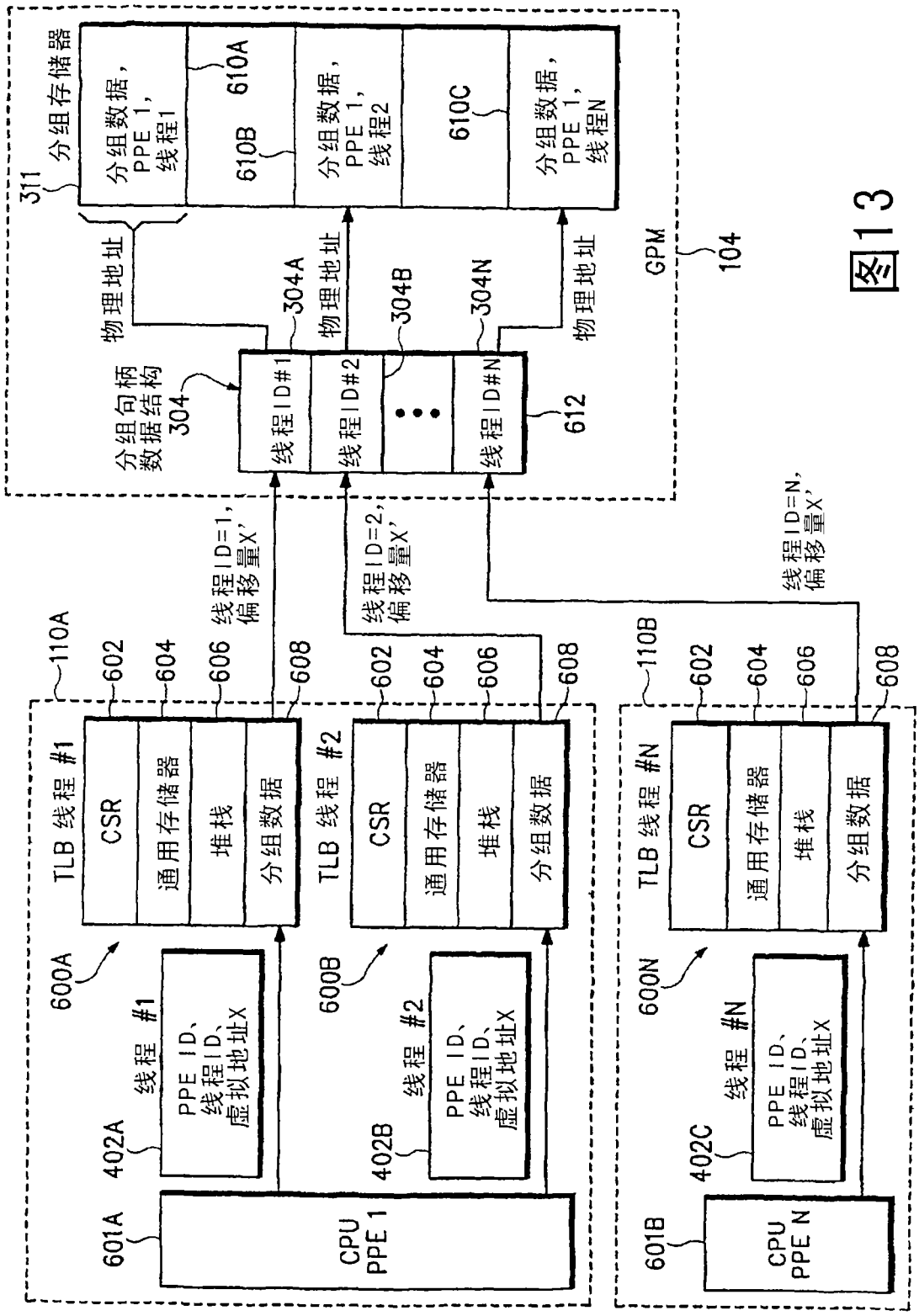

311
分组存储器
分组数据，PPE 1，线程1
610A
610B
分组数据，PPE 1，线程2
610C
分组数据，PPE 1，线程N
物理地址
304A
304B
304N
GPM
104
分组句柄数据结构
304
线程ID#1
线程ID#2
线程ID#N
612
线程ID=1，偏移量X'
线程ID=2，偏移量X'
线程ID=N，偏移量X'
110A
110B
602
604
606
608
TLB 线程 #1
TLB 线程 #2
TLB 线程 #N
CSR
通用存储器
堆栈
分组数据
600A
600B
600N
线程 #1
线程 #2
线程 #N
PPE ID、线程ID、虚拟地址X
402A
402B
402C
601A
601B
CPU PPE 1
CPU PPE N

图13

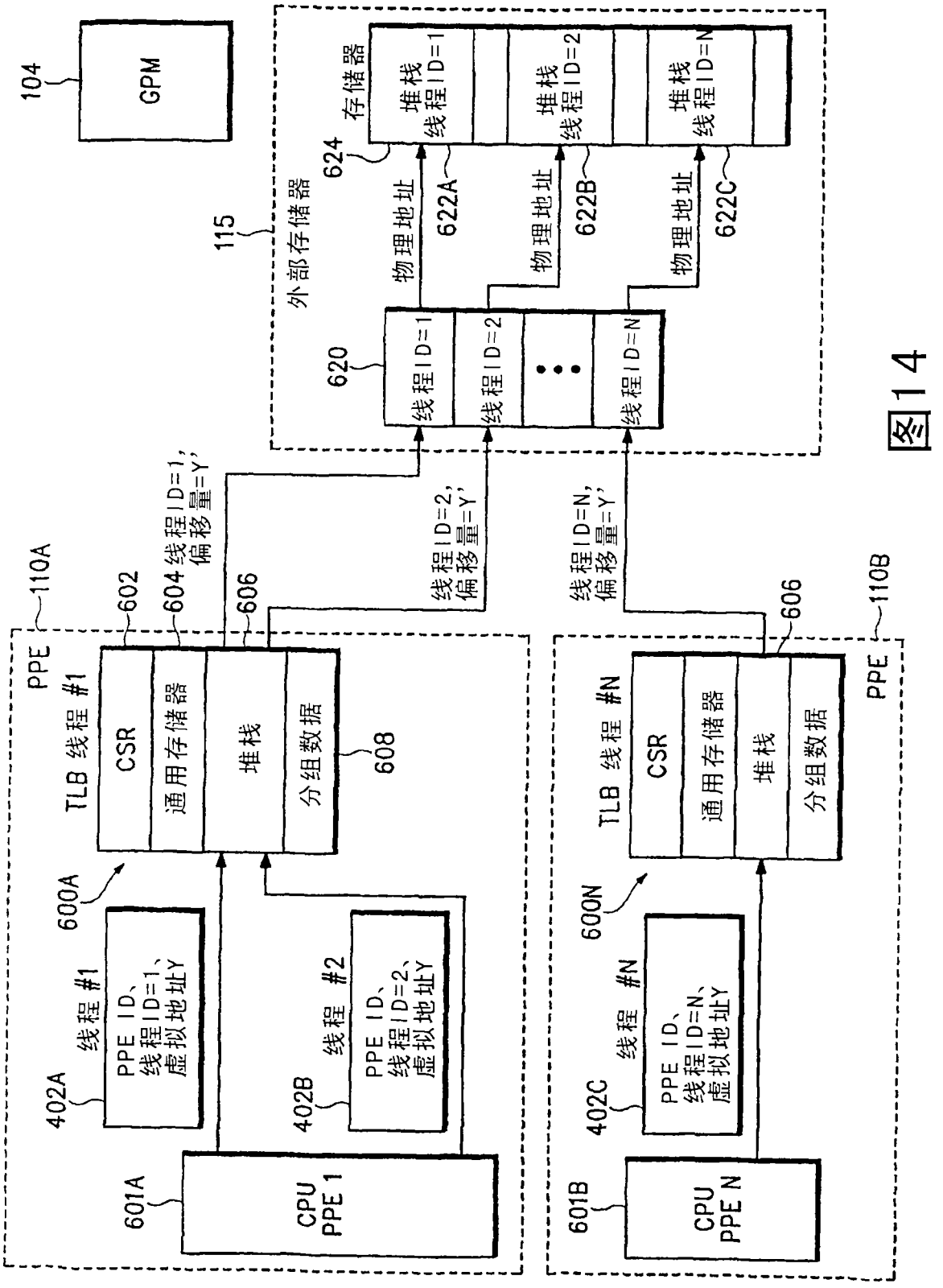

104
GPM
115
外部存储器
624
存储器
堆栈 线程ID=1
堆栈 线程ID=2
堆栈 线程ID=N
物理地址
622A
622B
622C
620
线程ID=1
线程ID=2
线程ID=N
线程ID=1, 偏移量=Y'
线程ID=2, 偏移量=Y'
线程ID=N, 偏移量=Y'
602
604
606
PPE
110A
110B
TLB 线程 #1
TLB 线程 #N
CSR
通用存储器
堆栈
分组数据
608
600A
600N
线程 #1
线程 #2
线程 #N
PPE ID、线程ID=1、虚拟地址Y
PPE ID、线程ID=2、虚拟地址Y
PPE ID、线程ID=N、虚拟地址Y
402A
402B
402C
601A
601B
CPU PPE 1
CPU PPE N

图14

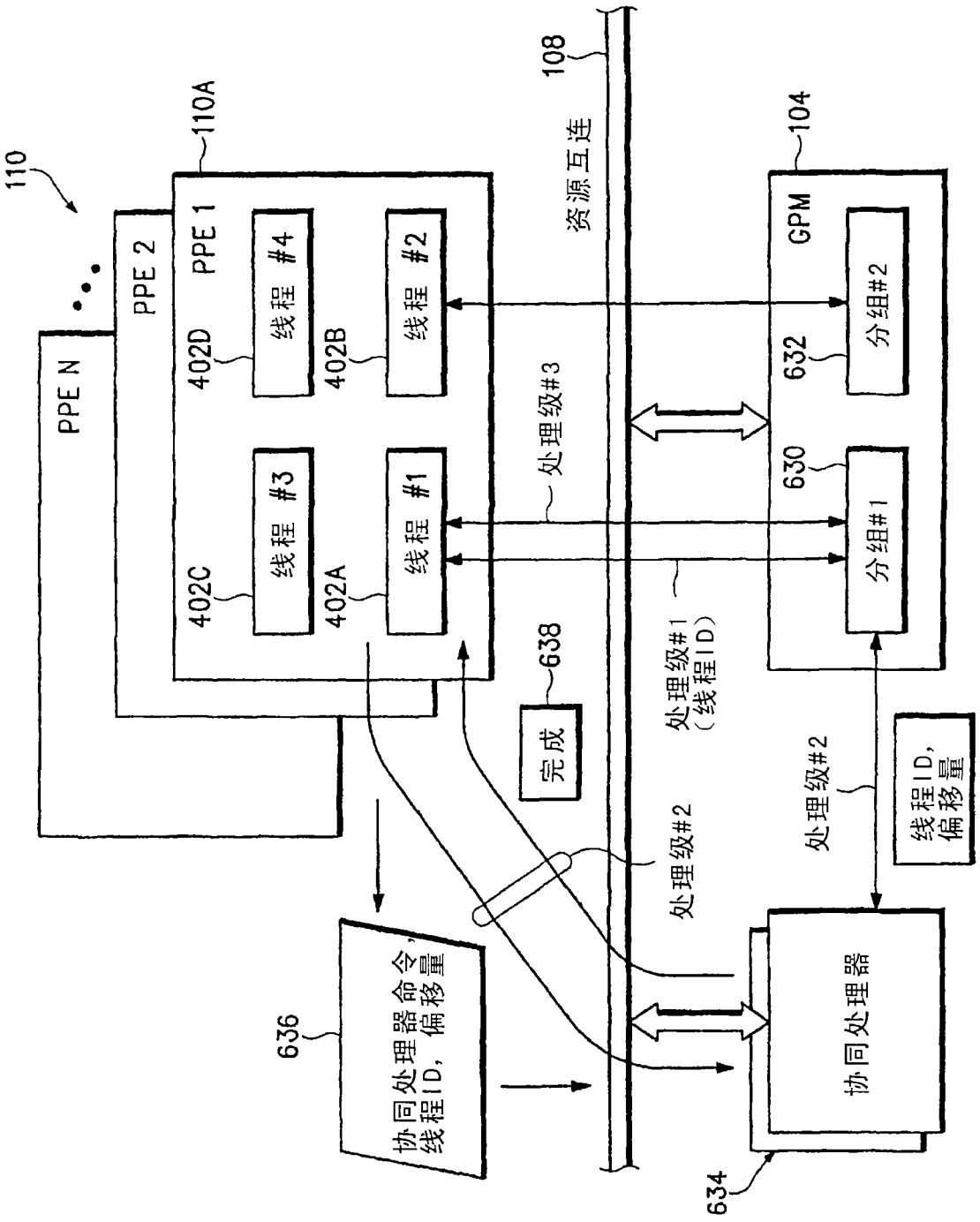

110
PPE N
PPE 2
PPE 1
110A
402D
线程 #4
402B
线程 #2
402C
线程 #3
402A
线程 #1
处理级#3
资源互连
108
104
GPM
632
分组#2
630
分组#1
638
完成
处理级#1
（线程ID）
处理级#2
处理级#2
线程ID，
偏移量
636
协同处理器命令，
线程ID，偏移量
协同处理器
634

图15

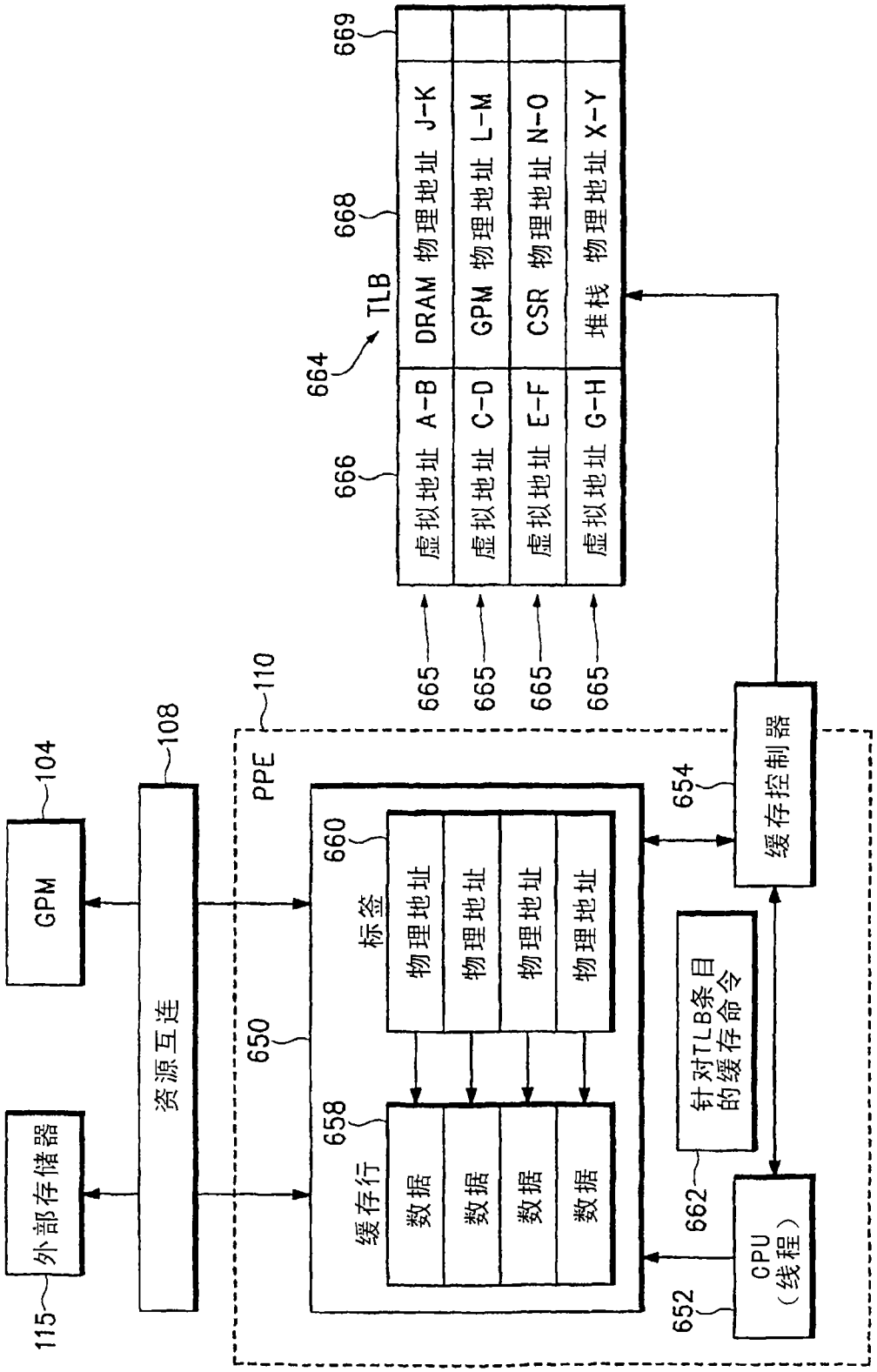

669
668
TLB
664
666
虚拟地址 A-B
虚拟地址 C-D
虚拟地址 E-F
虚拟地址 G-H
DRAM 物理地址 J-K
GPM 物理地址 L-M
CSR 物理地址 N-O
堆栈 物理地址 X-Y
665
665
665
665
110
PPE
108
104
GPM
资源互连
外部存储器
115
650
660
标签
物理地址
物理地址
物理地址
物理地址
658
缓存行
数据
数据
数据
数据
654
缓存控制器
针对TLB条目的缓存命令
662
CPU（线程）
652

图16

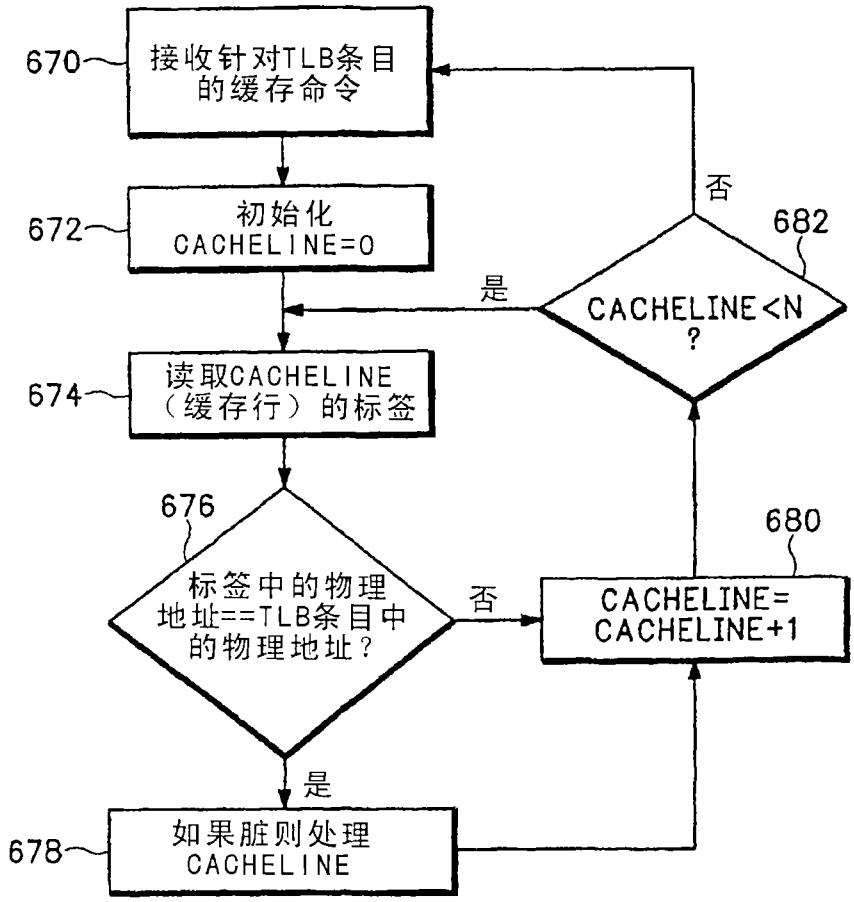

670
接收针对TLB条目的缓存命令
672
初始化 CACHELINE=0
674
读取CACHELINE（缓存行）的标签
676
标签中的物理地址==TLB条目中的物理地址？
否
是
678
如果脏则处理 CACHELINE
680
CACHELINE= CACHELINE+1
682
CACHELINE<N ?
是
否

图17