[19] 中华人民共和国国家知识产权局

[51] Int. Cl[7]
H04L 12/28

[12] 发明专利申请公开说明书

[21] 申请号 03814522.7

[43] 公开日 2005 年 8 月 31 日

[11] 公开号 CN 1663188A

[22] 申请日 2003.4.25 [21] 申请号 03814522.7

[30] 优先权

[32] 2002. 4.26 [33] GB [31] 0209670.9

[86] 国际申请 PCT/US2003/014259 2003.4.25

[87] 国际公布 WO2003/091857 英 2003.11.6

[85] 进入国家阶段日期 2004.12.21

[71] 申请人 美商传威股份有限公司

地址 美国康涅狄格州

[72] 发明人 K·德福尔彻 G·费尔布鲁根
L·德科斯特尔 J·沃特尔斯

[74] 专利代理机构 中国专利代理(香港)有限公司

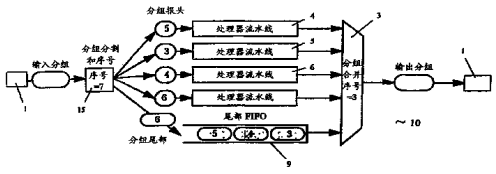
代理人 杨 凯 王 勇

权利要求书 6 页 说明书 30 页 附图 15 页

[54] 发明名称 有效的分组处理流水线装置和方法

[57] 摘要

一种在分组交换网络用于处理数据分组的分组处理设备包括：接收分组的装置；将管理信息添加到分组第一数据部分的装置，管理信息至少包括要应用到所述第一数据部分上的至少一个过程的指示；以及多条并行流水线，每条流水线包括至少一个处理单元，其特征在于：所述处理单元对所述第一数据部分执行所述管理信息指示的所述过程，以提供修改的第一数据部分。 根据方法，由每个处理单元执行的任务组织成多个函数，以便实际上只有函数调用而无函数间调用，并且一个处理单元的所述函数调用所调用的每个函数终止时，唯一的上下文是第一数据部分。



ISSN1008-4274

1. 一种在分组交换网络中使用的分组处理单元，包括：

用于在所述分组处理单元中接收分组的装置；

用于将管理信息添加到所述分组的至少第一数据部分中的装置，所述管理信息至少包括要应用到所述第一数据部分上的至少一个过程的指示；多条并行流水线，每条流水线包括至少一个处理单元，以及至少一个处理单元对所述第一数据部分执行所述管理信息指示的所述过程，以提供修改的第一数据部分。

2. 如权利要求 1 所述的分组处理单元，其特征在于还包括：将所述分组处理单元接收的每个分组分割成第一数据部分和第二数据部分的模块。

3. 如权利要求 1 或 2 所述的分组处理单元，其特征在于还包括：将所述修改的第一数据部分传送到另一处理单元的装置。

4. 如权利要求 3 所述的分组处理单元，其特征在于：所述传送装置只在所述管理信息指示的所述过程完成后将所述第一数据部分传送到另一处理单元。

5. 如权利要求 2 到 4 中任意一项所述的分组处理单元，其特征在于还包括：临时存储所述第二数据部分的装置。

6. 如权利要求 5 所述的分组处理单元，其特征在于：所述临时存储装置是 FIFO 存储单元。

7. 如以上任一权利要求所述的分组处理单元，其特征在于还包括：将所述接收分组的顺序指示添加到所述第一和第二数据部分中的装置。

8. 如以上任一权利要求所述的分组处理单元，其特征在于：每条流水线包括多个通信引擎，每个通信引擎链接到一个处理单元。

9. 如权利要求 8 所述的分组处理单元，其特征在于：每个通信引擎通过双端口存储单元链接到一个处理单元，其中一个端口连接

到所述通信引擎，而另一端口连接到所述处理单元。

10. 如权利要求 9 所述的分组处理单元，其特征在于：所述双端口存储器相对于相连的所述通信引擎配置为 FIFO。

11. 如以上任一权利要求所述的分组处理单元，其特征在于还包括：重新组合分组的所述第一和第二数据部分的重组单元。

12. 如权利要求 8 到 11 中任意一项所述的分组处理单元，其特征在于：所述通信引擎为处理单元选择分组的第一数据部分以进行处理。

13. 如权利要求 8 到 12 中任意一项所述的分组处理单元，其特征在于：处理单元的共享资源请求由所述通信引擎发送到共享资源。

14. 一种在分组交换网络所用分组处理设备中处理数据分组的方法，所述分组处理单元包括多条并行流水线，每条流水线包括至少一个处理单元；所述方法包括：

将管理信息添加到所述分组的至少第一数据部分中，所述管理信息至少包括要应用到所述第一数据部分上的至少一个过程的指示；以及

使用至少一个处理单元，对所述第一数据部分执行所述管理信息指示的所述过程，以提供修改的第一数据部分。

15. 如权利要求 14 所述的方法，其特征在于还包括：将所述分组处理单元接收的每个分组分割成第一数据部分和第二数据部分。

16. 如权利要求 14 或 15 所述的方法，其特征在于还包括：将所述修改的第一数据部分传送到另一处理单元。

17. 如权利要求 16 所述的方法，其特征在于：所述传送步骤包括只在所述管理信息指示的所述过程完成后将所述第一数据部分传送到另一处理单元。

18. 如权利要求 15 到 17 中任意一项所述的方法，其特征在于还包括：临时存储所述第二数据部分。

19. 如权利要求 18 所述的方法，其特征在于：所述临时步骤包

括存储在 FIFO 存储单元中。

20. 如权利要求 15 到 19 所述的方法，其特征在于还包括：将所述接收分组的顺序指示添加到所述第一和第二数据部分中。

21. 如权利要求 15 到 20 中任意一项所述的方法，其特征在于还包括：重新组合分组的所述第一和第二数据部分。

22. 一种在分组交换网络中使用的分组处理单元，包括：

在所述分组处理单元中接收分组的装置；

将所述分组处理单元接收的每个分组分割成第一数据部分和第二数据部分的模块；

处理至少所述第一数据部分的装置；以及

重新组合所述第一和第二数据部分的装置。

23. 如权利要求 22 所述的分组处理单元，其特征在于：所述分组处理单元包括多条并行流水线，每条流水线包括至少一个处理单元；每个处理单元执行的任务组织成多个函数，以便实质上只有函数调用而无函数间调用，并且一个处理单元的所述函数调用所调用的每个函数终止时，唯一的上下文是第一数据部分。

24. 如权利要求 22 所述的分组处理单元，其特征在于还包括：

用于将管理信息添加到所述分组的至少第一数据部分中的装置，所述管理信息至少包括要应用到所述第一数据部分上的至少一个过程的指示；以及

多条并行流水线，每条流水线包括至少一个处理单元，至少一个处理单元对所述第一数据部分执行所述管理信息指示的所述过程，以提供修改的第一数据部分。

25. 如权利要求 24 所述的分组处理单元，其特征在于还包括：将所述处理的第一数据部分传送到另一处理单元的装置。

26. 如权利要求 25 所述的分组处理单元，其特征在于：所述传送装置只在所述管理信息指示的所述过程完成后将所述第一数据部分传送到另一处理单元。

27. 如权利要求 22 到 26 中任意一项所述的分组处理单元，其特征在于还包括：临时存储所述第二数据部分的装置。

28. 如权利要求 27 所述的分组处理单元，其特征在于：所述临时存储装置是 FIFO 存储单元。

29. 如权利要求 23 到 28 中任意一项所述的分组处理单元，其特征在于还包括：将所述接收分组的顺序指示添加到所述第一和第二数据部分中的装置。

30. 如权利要求 23 到 29 中任意一项所述的分组处理单元，其特征在于：每条流水线包括多个通信引擎，每个通信引擎链接到一个处理单元。

31. 如权利要求 30 所述的分组处理单元，其特征在于：每个通信引擎通过双端口存储单元链接到一个处理单元，其中一个端口连接到所述通信引擎，而另一端口连接到所述处理单元。

32. 如权利要求 31 所述的分组处理单元，其特征在于：所述双端口存储器相对于相连的所述通信引擎配置为 FIFO。

33. 一种在分组交换网络所用分组处理设备中处理数据分组的方法，包括将所述分组处理单元接收的每个分组分割成第一数据部分和第二数据部分；

处理所述至少第一数据部分；以及

重新组合所述第一和第二数据部分。

34. 如权利要求 33 所述的方法，其特征在于：所述分组处理单元包括多条并行流水线，每条流水线包括至少一个处理单元；所述方法还包括：将每个处理单元执行的任务组织成多个函数，以便实质上只有函数调用而无函数间调用，并且一个处理单元的所述函数调用所调用的每个函数终止时，唯一的上下文是第一数据部分。

35. 如权利要求 33 所述的方法，其特征在于：所述分组处理单元包括多条并行流水线，每条流水线包括至少一个处理单元；所述方法还包括：

将管理信息添加到所述分组的至少第一数据部分中，所述管理信息至少包括要应用到所述第一数据部分上的至少一个过程的指示；以及

至少一个处理单元对所述第一数据部分执行所述管理信息指示的所述过程，以提供修改的第一数据部分。

36. 如权利要求 35 所述的方法，其特征在于：将所述处理的第一数据部分传送到另一处理单元。

37. 如权利要求 36 所述的方法，其特征在于：所述传送步骤包括只在所述管理信息指示的所述过程完成后将所述第一数据部分传送到另一处理单元。

38. 如权利要求 33 到 37 中任意一项所述的方法，其特征在于还包括：临时存储所述第二数据部分。

39. 如权利要求 38 所述的方法，其特征在于：所述临时步骤包括存储在 FIFO 存储单元中。

40. 如权利要求 33 到 39 所述的方法，其特征在于还包括：将所述接收分组的顺序指示添加到所述第一和第二数据部分中。

41. 一种在分组交换网络中使用的分组处理单元，包括：

用于在所述分组处理单元中接收分组的装置；

多条并行流水线，每条流水线包括至少一个处理单元、通过双端口存储单元链接到所述至少一个处理单元的通信引擎，其中一个端口连接到所述通信引擎，而另一端口连接到所述处理单元。

42. 如权利要求 41 所述的移动站，其特征在于：所述双端口存储器相对于相连的所述通信引擎配置为 FIFO。

43. 一种在分组交换网络所用分组处理设备中处理数据分组的方法，其特征在于：所述分组处理单元包括多条并行流水线，每条流水线包括至少一个处理单元；所述方法还包括：将每个处理单元执行的任务组织成多个函数，以便实质上只有函数调用而无函数间调用，并且一个处理单元的所述函数调用所调用的每个函数终止时，

唯一的上下文是第一数据部分。

44. 一种在分组交换网络中使用的分组处理单元，包括:

用于在所述分组处理单元中接收分组的装置;

多条并行流水线，每条流水线包括用于对至少部分数据分组执行某种过程的至少一个处理单元;

连接到所述处理单元的通信引擎;

至少一个共享资源，其中：所述通信引擎适于从所述处理单元接收共享资源请求，并将它发送到所述共享资源。

有效的分组处理流水线装置和方法

发明领域

本发明涉及电信网络，具体地说，涉及分组交换电信网络，更具体地说，涉及其中所用的网元和通信模块，以及例如在网络节点上操作所述网元和通信模块以处理分组的方法。

现有技术

以确定和灵活的方式处理以高速率到达例如电信网络节点的分组，优选这样的体系结构：在考虑灵活的处理单元如处理器内核的同时，将处理分组的特殊性纳入考虑。分组处理的理想属性是处理分组过程中固有的并行性、数据平面和控制平面（单个处理线程可在其上停止）两者中的高 I/O（输入/输出）要求以及需要尽可能有效地加以使用的极小周期预算。并行处理对于高吞吐量的分组交换电信网络中的分组处理是有利的，可提高处理能力。

虽然处理可并行执行，但某些需要访问的资源却不无备份。这导致不止一个处理单元希望访问此类资源。例如数据库等共享资源是可由多个处理单元访问的一个资源。每个处理单元可执行单独的任务，该任务不同于任何其它处理单元执行的任务。作为任务的一部分，访问共享资源可能是必需的，例如，访问数据库以获得相关的内联数据。尝试使吞吐量最大时，访问处理单元的共享资源通常具有长的等待时间。如果处理单元在从共享资源收到应答前暂停操作，则效率变得很低。而且，需要大存储空间的资源一般不在芯片上，因此，访问和检索时间相当大。

传统上，对具有例如处理器内核的处理单元上的处理进行优化会涉及上下文切换，即，暂停一个线程，并将存储在寄存器中的所有当前数据保存到存储器中，这样，以后在从共享资源收到应答时，

可重新创建相同的上下文。然而，上下文切换占用大量的处理器资源，或者在只为此任务分配了少量处理器资源时占用大量的时间。

本发明的目的是提供一种分组处理单元和以提高的效率操作所述单元的一种方法。

本发明的又一目的是提供一种分组处理单元和操作所述单元的一种方法，通过本发明，上下文切换具有低处理时间开销和/或低处理资源分配。

本发明的又一目的是提供一种高效的分组处理单元和使用并行处理操作所述单元的方法。

发明概述

本发明解决了此问题，并在保持简单的编程模型的同时取得了很高的效率，而同时在处理单元上不需要代价很高的多线程处理并可以调整处理单元以适应某个特殊功能。本发明部分依赖于如下事实：对于上下文切换，当分组交换电信网络的网元中发起共享资源请求时，通常存在用途很少的上下文，或者可以通过明智的任务编程将有用的上下文降到最少。切换以处理另一分组并不一定要求保存处理单元的完整状态。明智的编程可包括将要在每个处理单元上运行的程序组织为函数调用序列，每次调用在处理单元上运行时具有上下文，但除了在分组本身中的数据外，不需要函数间调用。

因此，本发明提供了一种在分组交换网络所用分组处理设备中处理数据分组的方法，所述分组处理设备包括多条并行流水线，每条流水线包括用于处理一个数据分组的一部分的至少一个处理单元；所述方法还包括：将每个处理单元执行的所述任务组织成多个函数，这样，实质上只存在函数调用而无函数间调用，并且在一个处理单元的所述函数调用所调用的每个函数终止时，唯一的上下文是第一数据部分。

本发明提供了在分组交换网络中使用的分组处理设备，所述分

组处理设备包括：用于在所述分组处理设备中接收分组的装置；用于将管理信息添加到所述分组的至少第一数据部分中的装置，所述管理信息至少包括要应用到所述第一数据部分上的至少一个过程的指示；多条并行流水线，每条流水线包括至少一个处理单元，并且所述至少一个处理单元对所述第一数据部分执行所述管理信息指示的所述过程，以提供修改的第一数据部分。

本发明还提供了在分组处理设备中使用的通信模块，所述通信模块包括：用于在所述通信模块中接收分组的装置；用于将管理信息添加到所述分组的至少第一数据部分中的装置，所述管理信息至少包括要应用到所述第一数据部分上的至少一个过程的指示；多条并行通信流水线，每条通信流水线与至少一个处理单元一起使用；以及用于存储所述第一数据部分的存储装置。

本发明还提供了在分组交换网络所用分组处理设备中处理数据分组的方法，所述分组处理设备包括多条并行流水线，每条流水线包括至少一个处理单元；所述方法包括：将管理信息添加到所述分组的至少第一数据部分中，所述管理信息至少包括要应用到所述第一数据部分上的至少一个过程的指示；以及所述至少一个处理单元对所述第一数据部分执行所述管理信息指示的所述过程，以提供修改的第一数据部分。

本发明还提供了在分组交换网络中使用的分组处理设备，所述分组处理设备包括：用于在所述分组处理设备中接收分组的装置；用于将所述分组处理设备接收的每个分组分割成第一数据部分和第二数据部分的模块；用于处理所述至少第一数据部分的装置；以及重新组合所述第一和第二数据部分的装置。

本发明还提供了在分组交换网络所用分组处理设备中处理数据分组的方法，所述包括：将所述分组处理设备接收的每个分组分割成第一数据部分和第二数据部分；处理所述至少第一数据部分；以及重新组合所述第一和第二数据部分。

本发明还提供了在分组交换网络中使用的分组处理设备，所述分组处理设备包括：用于在所述分组处理设备中接收分组的装置；多条并行流水线，每条流水线包括至少一个处理单元、通过双端口存储单元链接到所述至少一个处理单元的通信引擎，其中一个端口连接到所述通信引擎，而另一端口连接到所述处理单元。

本发明还提供了在分组处理设备中使用的通信模块，所述通信模块包括：用于在所述通信模块中接收分组的装置；多条并行通信流水线，每条通信流水线包括与处理单元通信以便处理分组的至少一个通信引擎和双端口存储单元，所述双端口存储单元的一个端口连接到所述通信引擎。

本发明还提供了在分组交换网络中使用的分组处理单元，所述分组处理单元包括：用于在所述分组处理单元中接收数据分组的装置；多条并行流水线，每条流水线包括用于对数据分组的至少一部分执行某种过程的至少一个处理单元、连接到所述处理单元的通信引擎以及至少一个共享资源，其中：所述通信引擎适于从所述处理单元接收共享资源请求，并将它发送到所述共享资源。所述通信引擎还适于从所述共享资源接收应答。

本发明还提供了配合分组处理单元使用的通信模块，所述通信模块包括：用于在所述通信模块中接收数据分组的装置；多条并行流水线，每条流水线包括至少一个通信引擎以及至少一个共享资源，所述通信引擎具有用于连接到处理单元的装置，其中：所述通信引擎适于接收共享资源请求，并将它发送到所述共享资源，以及从所述共享资源接收应答并将它发送到所述连接装置以便连接到所述处理单元。

现在将参照如下附图描述本发明。

附图简述

图 1a 和 1b 显示了根据本发明实施例的分组处理路径；

图 2a 和 2b 显示了根据本发明实施例的分组调度操作;

图 3 显示了根据本发明实话例的一条流水线细节;

图 4a 显示了与根据本发明实施例的处理单元相关联的 FIFO 存储器中报头的位置;

图 4b 显示了根据本发明实施例的报头;

图 5 显示了根据本发明实施例的处理单元;

图 6 显示了根据本发明实施例如何通过流水线处理分组;

图 7 显示了根据本发明实施例，传送期间的分组重新对齐;

图 8 显示了根据本发明实施例的通信引擎;

图 9 显示了根据本发明实施例，用于控制缓冲区中报头队列的指针排列;

图 10 显示了根据本发明又一实施例的共享资源布置;

图 11 显示了根据本发明处理分组报头（packet head）的流程图。

说明性实施例的详细说明

下面将参照某些实施例和附图描述本发明，但本发明并不限于此。本领域的技术人员会理解，本发明在并行处理领域和/或电信网络，特别是分组交换电信网络的分组处理中具有广泛应用。

本发明的一个方面是可在分组处理设备中用于分组标题处理的分组处理通信模块。分组处理设备由多条处理流水线组成，而每条流水线由若干处理单元组成。处理单元包括处理部件，例如处理器和相关联的存储器。处理器可以是微处理器，或者可以是可编程数字逻辑单元，例如可编程阵列逻辑（PAL）、可编程逻辑阵列（PLA）、可编程门阵列，特别是现场可编程逻辑阵列。分组处理通信模块包括流水式通信引擎，该通信引擎提供适用于处理单元的非本地通信功能。为了形成完整的分组处理设备，将处理器内核和可选的其它处理功能块安装在分组处理通信模块上。处理器内核无需具有内置的本地硬件上下文切换功能。

在下述内容中，将主要针对完整的分组处理设备描述本发明，但应理解，与根据本发明的分组处理通信模块一起使用的处理器内核类型和尺寸不一定限制本发明，并且通信模块（无处理器）也是本发明的一个独立方面。

本发明的一个方面是优化的软件/硬件划分。例如，处理单元最好与负责非本地通信，称为通信引擎的硬件块相组合。该硬件块可以常规方式实现，例如，实现为逻辑阵列如门阵列。然而，本发明还可通过替代装置来实现，例如，通信引擎可实现为可配置块，如可通过使用可编程数字逻辑单元，如可编阵列逻辑（PAL）、可编程逻辑阵列（PLA）、可编程门阵列，特别是现场可编程逻辑阵列获得的可配置块。具体地说，为了尽快地提供产品，本发明包括了两代或更多代的智能设计策略，由此第一代中使用的可编程装置在后续代中为专用硬件块所替代。

硬件块最好用于协议无关的功能。对于协议相关的功能，最好采用协议改变时允许重新配置和重新设计的软件块。例如，微处理器对此类应用有利。

根据本发明实施例的完整分组处理设备 10 包括安装有处理器的分组处理通信模块。处理设备 10 具有如图 1a 所示，由多条并行处理流水线 4、5、6 组成的分组处理路径。流水线的数量取决于要实现的处理能力。如图 1b 所示，处理路径包括用于从例如电信网络 1 接收分组并将分组分发到一条或多条并行处理流水线 4、5、6 的调度单元 2。电信网络 1 可以是任何分组交换网络，例如，陆线或移动无线电电信网络。每个接收分组包括标题和有效负荷。每条流水线 4、5、6 包括多个处理单元 4b...e、5b...e、6b...e。这些处理单元适于处理至少分组的标题。分组处理单元 4b...e、5b...r、6b...e 可与诸如数据库等太大（或昂贵）而无法为每个处理单元（例如，路由表）复制的若干其它电路部件连接。同样地，一些信息需要由多条流水线更新或采样（例如，统计信息或管制信息）。因此，可添加处理单

元可以与其通信的多个所谓的共享资源 SR1-SR4。根据本发明的一个方面，提供了特定的通信基础结构以便处理单元与共享资源通信。由于共享资源可以远离处理单元，并且由于它们处理多个处理器的请求，因此，请求与应答之间的等待时间会很长。具体地说，处理单元 4b...e、5b...e、6b...e 中的至少一个单元可以经单条总线 8a、8b、8c、8d、8e 和 8f 访问一个或多个共享资源，例如，处理单元 4b、5b、6b 经总线 8a 访问 SR1，处理单元 4b、5b、6b 和 4c、5c、6c 以及 4e、5e、6e 分别经总线 8b、8c 和 8d 访问 SR2。总线 8 可以是任何适用的总线，并且该总线的形式不视为对本发明的限制。入口分组缓冲区 4a、5a、6a 和/或出口分组缓冲区 4f、5f、6f 可分别设在处理流水线之前和/或之后。分组缓冲区的一个功能可以是适应数据路径带宽。分组缓冲区的主要任务是将主数据路径通信带宽从网络 1 转换到流水线通信带宽。除此之外，分组缓冲区中可提供一些其它功能，如开销插入/去除和任务查找。分组缓冲区最好能够缓冲单个报头（它包括至少分组标题）。它确保在接收和发送侧上对与一个报头一样大的突发的线速数据传送。

如图 1a 中所示，例如从通信网络 1 传来的输入分组被分割和序号分配装置分割成报头和尾部，该装置最好在调度单元 2 中实施。报头包括分组标题，而尾部包括至少部分分组有效负荷。报头馈入流水线 4-6 之一，而有效负荷存储(缓冲)到合适的存储装置 9 如 FIFO 中。在处理后，报头和有效负荷在重组单元 3 中重新组合（分组合并），然后输出，例如，在通过网络 1 发送到另一节点前，可以在其中进行缓冲。

通常，一个或多个共享资源 SR1-4 可用于处理路径，为流水线中的处理单元处理特定的任务。例如，这些共享资源可以是使用芯片外资源中存储的数据结构的专用查找引擎，或者是用于需要访问共享信息的专用功能的专用硬件。本发明在如下情形中对于提高效率特别有利：如果每个处理单元暂停直至相关共享资源作出响应，

而这些要用于处理系统中的共享资源引擎响应请求的等待时间很长，这种等待时间导致流水线处理单元效率下降。可用于本发明的典型共享资源是 IP 转发表、MPLS 转发表、管制数据库、统计信息数据库。例如，由共享资源辅助的流水线结构执行的功能可以为：

- IPv4/IPv6 标题分析和转发
- 多字段分类
- MPLS 标签分析与交换
- IPinIP 或 GRE 隧道端接
- MPLS 隧道端接
- IPinIP 或 GRE 隧道封装
- MPLS 隧道封装
- 计量与统计信息收集
- ECMP 和中继支持
- QoS 模型支持

为此，流水线结构可由以下共享资源辅助：

- 32 位或 128 位最长前缀匹配单元
- TCAM 分类装置
- 芯片外 DRAM、芯片外 SRAM、芯片上 SRAM
- 6B 或 18B 精确匹配单元
- 32 位或 128 位源滤波器（最长前缀匹配单元）
- 计量单元

使用共享资源的一个方面是在等待对发送到共享资源的请求的应答时处理单元的停止时间。为使处理单元放弃一个当前待处理的任务，切换到另一任务，然后返回第一个任务，通常是提供上下文切换，即存储处理器单元寄存器的内容。本发明的一个方面是采用硬件加速上下文切换。这也允许处理器内核用于自身未配备硬件切换功能的处理单元。此硬件最好在每个处理节点提供，例如，以通信引擎的形式提供。每个处理单元维护要处理的分组池。在发出共

享资源请求时，相关处理单元的处理装置将上下文切换到另一分组，直至已收到该请求的应答。本发明的一个方面是利用分组处理并行性，这样，可以尽可能高的效率使用处理单元，进行有用的处理，从而避免等待 I/O（输出/输出）操作完成。例如，这些 I/O 操作是请求共享资源或者将分组信息拷贝到处理单元内或从中拷贝出。本发明部分依赖于如下事实：当分组交换电信网络的网元中发起共享资源请求时，通常存在用途很少的上下文，或者可以通过明智的任务编程将有用的上下文降到最少。切换以处理另一分组并不一定要求保存处理单元的完整状态。明智的编程可包括将要在每个处理单元上运行的程序组织为函数调用序列，每次调用在处理单元上运行时具有上下文，但不需要函数间调用。例外是分组本身或分组的一部分中数据提供的上下文。

回到图 1a 和 1b 及分割装置 15，报头的大小经过选择，以便它包含与分组一起接收的所有相关标题。例如，这可以通过在分组的固定点进行分割来完成（在所支持的最大尺寸的标题后）。这会导致一些有效负荷被分割到报头中。由于通常不处理有效负荷，因此这一般不会有问题。然而，本发明包括了处理有效负荷的可能性，例如，用于网络速率控制。在分组数据包含可多重分解的数据时，在允许的情况下，视节点的网络转发带宽而定，网络可将数据截断，从而具有更低的分解度。为处理此类情况，本发明在其范围内包括了更精确的分组评估，以识别标题和有效负荷，并在其结合处干净地将它们分割。分离的报头（或标题）馈入处理流水线，而尾部（或有效负荷）则加以缓冲（及采用未显示的其它处理单元加以任意处理）并在处理后重新附加到（修改的）报头。

在分割后，接着将报头提供给一条处理流水线，而将尾部存储在诸如 FIFO 9 的存储器中。每个分组最好由序号分配模块 15 分配一个序号。随后将此序号复制到报头及每个分组的尾部中并予以存储。所述序号具有如下三个用途：

- 在流水线末重新装配（修改后）报头与尾部
- 需要时删除报头及其对应的尾部
- 需要时保持分组的特定顺序

序号可由分组分割和序号分配装置 15 中包括的计数器生成。序计数器随每个输入分组递增。这样，序号可用于在流水线末以特定顺序放置分组。

开销生成器设置在分组调度器 2 中，或者最好设置在分组缓冲区 4a、5a、6a 中，以便为每个报头和/或尾部生成新的/附加的开销。在生成完整的报头后，可将报头发送到具有可用缓冲空间的流水线 4-6 之一。尾部发送到尾部 FIFO 9。

根据本发明的实施例，增加的开销包括在报头和/或尾部中的管理数据。图 2a 中示意性地显示了处理流程。在尾部，新的开销最好包含序号和长度，即有效负荷的长度，并可选地包括对用于处理对应报头的流水线的引用。在报头中，增加的开销最好包括报头管理字段（HAF）和存储由分组处理流水线生成的结果和状态的区域。因此，报头可包括结果存储、状态存储和管理数据存储。HAF 可包含报头长度、偏移、序号和执行 FIFO 维护和报头选择所必需的多个字段。

图 2b 显示了对处理设备内的分组执行的另一组操作。可在由流水线处理的每个报头前面加上可用于存储中间结果的临时暂存区。它也可用于构建分组描述符，分组描述符可由分组处理单元的下游处理装置使用。如图 2b 所示，每条流水线开始处的分组缓冲区 4a、5a、6a 可将此临时暂存区添加到分组报头中。在末端的分组缓冲区 4f、5f、6f 可删除它（至少部分删除）。分组进入分组处理单元时，标题包含定义分组协议的一些链路层信息。这要由分组处理单元转换成指向要对分组执行的第一个任务的指针。此查找操作可由入口分组缓冲区 4a、5a、6a 执行。

本发明的一个方面是，报头在流水线中时包括要由当前和/或下

一处理单元执行的任务引用。这样，处理器单元上下文的一部分存储在报头中。也就是说，报头中 HAF 的当前版本等效于处理状态，包括要对该报头执行的下一过程的指示。报头本身还可存储内联数据，例如，变量的中间值可存储在临时暂存区中。为处理单元提供其上下文所必需的所有信息因此存储在报头中。报头沿流水线下移时，上下文以存储在报头相关部分，如 HAF、临时暂存区中的数据的形式随报头一起移动。因此，本发明的一个创新方面是上下文随分组一起移动，而不是上下文相对于某个处理器不动。

分组重组模块 3 重新组合从处理流水线 4-6 传来的分组报头和从尾部 FIFO 9 传来的对应尾部。分组网络可划分割成可在每个节点上对每个分组进行独立路由的网络（数据报网络）和其中建立了虚拟电路并且源与目的地之间的分组使用这些虚拟电路之一的网络。因此，视网络而定，对分组排序可能有不同的要求。重组模块 3 确保分组以它们的到达顺序离开，或以所需的任何其它顺序离开。分组重组模块 3 具有跟踪发送的最后分组的序号的装置。它搜索不同处理流水线的输出以查找具有可发送序号的报头，以及搜索 FIFO 9 的末端以查看可用于传输的尾部，例如，下一序号。为简化操作，最好是在流水线中严格根据序号处理分组，这样，报头及其对应尾部在重组模块 3 中同时可用。因此，最好是用于处理流水线中分组的装置严格根据提供的序号操作。随后，在适当的报头传播到流水线输出后，会在重组模块 3 中将其添加到对应的尾部中，该尾部最好是那时尾部 FIFO 9 中的第一个条目。重组单元 3 或出口分组缓冲区 4f、5f、6f 从报头中删除剩余的 HAF 和其它字段。

分组必须丢弃时，处理单元具有用于在报头中设置要丢弃报头的指示的装置，例如，它可以在分组开销中设置丢弃标志。重组模块 3 随后负责丢弃此报头和对应的尾部。

图 3 示意性地显示了根据本发明实施例的一条流水线 4。分组报头最好在受到处理单元最小干预的情况下，沿多条总线从一个处理

级传递到另一个处理级。另外，处理单元需要能够在传送期间持续处理分组。每个处理单元 4b...4d 最好包括处理部件 14b-14d 和通信引擎 11b-11d。通信引擎可用硬件如可配置的数字逻辑单元来实现，并且处理单元可包括可编程处理内核，但本发明不限于此。为每个处理单元 4b-4d 分别分配了某种专用存储器。例如，每个处理部件的部分数据存储器最好是双端口存储器，例如，双端口 RAM 7b...7d 或类似存储器。其一个端口由通信引擎 11b...11d 使用，另一端口连接到此处理单元的处理部件。根据本发明的一个实施例，通信引擎 11b...11d 在一些情况下操作存储器 7b...7d 中存储的报头，仿佛此存储器组织为 FIFO 一样。为此，报头可以如同在 FIFO 中一样以逻辑或物理方式存储。通过这种方式，根据报头的到达顺序将报头压入此存储器和从中弹出。然而，视应用而定，通信引擎并不限于以此方式使用存储器 7b...7d，而是可以利用此存储器的所有功能，例如，将其作为双端口 RAM 使用。在处理报头时保持报头之间先入先出关系的优点是分组输入顺序将自动保持，从而产生相同的输出分组顺序。然而，本发明并不限于此，而是包括可由通信引擎以随机方式访问的数据存储器。

通信引擎彼此之间进行通信以传送报头。因此，在每个通信引擎准备接收新数据时，将就绪信号发送到前一通信引擎或其它以前的电路部件。

根据如图 4a 所示的本发明实施例，从 RAM 7b...7d 的输入端口到输出端口，设置了三个存储器区域：一个包含已处理并准备发送到下一级的报头，另一个包含正在处理的报头，第三个区域包含已部分接收但尚未准备处理的报头。RAM 7b...7d 被分割成多个大小相等的缓冲区 37a-h。每个缓冲区 37a-h 只包含一个报头。如图 4b 所示，每个报头包含：

- 报头管理字段（HAF）：HAF 包含分组管理所需的所有信息。它一般是一个 64 位字长。各缓冲区 37a-h 均具有存储

HAF 数据的装置。

- 临时暂存区：用作高速暂存存储器的可选区域，在处理器之间传递分组状态，或构建将离开系统的分组描述符。各缓冲区 37a-h 最好均具有在临时暂存区存储数据的装置。
- 分组开销：要从发组删除（去封装）或要添加到分组（封装）中的开销。各缓冲区 37a-h 最好均具有存储分组开销的装置。
- 报头分组数据：分组的实际报头数据。各缓冲区 37a-h 最好均具有存储报头分组数据的装置。
- 共享资源请求：除分组外，每个缓冲区在缓冲区末端为共享资源请求提供了一定空间。各缓冲区 37a-h 最好均具有存储共享资源请求的装置。

HAF 包含分组信息（长度）和处理状态以及包含“层 2”（存在时）的部分信息（例如，至少为指示物理接口类型的代码和“层 3”协议号）。

根据本发明实施例的通信模块可包括调度单元 2、分组重组单元 3、存储器 9、通信引擎 11b...d、双端口 RAM 7b-d、可选的分组缓冲区以及到处理单元和共享资源的适当连接点。当通信模块配备了相应的处理单元时，就形成了有效的分组处理设备。

图 5 示意性地显示了根据本发明实施例的处理单元。处理单元 4b 包括处理部件 14b、最好实现为双端口 RAM 的报头缓冲存储器 7b、程序存储器 12b 和通信引擎 11b。可提供用于处理部件的本地存储器 13b。程序存储器 12b 经指令总线连接到处理部件 14b，并用于存储在处理部件 14b 上运行的程序。缓冲存储器 7b 通过数据总线 17b 连接到处理部件 14b。通信引擎 11b 经监控总线 18b 监控数据总线，以检测从处理部件到缓冲区之一中的任一 HAF 的写入访问。这允许通信引擎 11b 监视并更新其内部寄存器中每个缓冲区的状态。通信引擎 11b 通过数据存储器总线 19b 连接到缓冲存储器 7b。可选地，一

个或多个处理块（未显示）可与处理部件 14b 包括在一起，例如，与诸如加密块的协处理装置包括在一起，以便降低处理部件 14b 的负荷，用以执行重复性的数据密集型任务。

使用处理内核如美国加州圣克拉拉 Tensilica 的 Xtensa®内核，可有效地实现根据本发明的处理部件 14b。带有专用硬件指令以加速将映射到此处理部件上的功能的处理器内核，在灵活性与性能之间取得了很好的折衷。另外，可在这种处理器内核中添加所需的处理部件硬件支持，即，处理器内核不需要上下文切换硬件支持。处理部件 14b 通过系统总线 20b 连接到通信引擎 11b－复位和中断可通过单独的控制总线传送（最好如图 8 中所示）。从处理部件的角度看，数据存储器 7b 不是 FIFO，而只是分组池，可采用多种不同的选择算法从中选择分组来进行处理。

根据本发明的一个方面，处理部件是同步的，使得缓冲区 37a-h 不会上溢或下溢。对报头的处理在处理部件上适时完成。分组一到达就从系统中移去，因此，处理将从不会产生对额外缓冲空间的需要。因此，处理部件应该不会产生缓冲区溢出。处理分组只可在足够的数据可用时开始。在无报头适合处理时，硬件（通信引擎）暂停处理部件。RAM 7b...7d 提供缓冲存储空间，并允许处理部件与流水线的处理步调无耦合。

每个处理部件可决定丢弃分组或剥离报头的一部分或将某些内容添加到报头中。为丢弃报头，处理单元只需在 HAF 中将丢弃标志置位。这将产生两种效果：报头将不再适合进行处理，并且只有该 HAF 将被传送到下一级。分组重组器 3 收到丢弃比特已置位的报头时，它会丢弃对应的尾部。

HAF 具有指示第一相关字节位置的偏移字段。在输入分组中，这将始终等于零。为从一开始就剥离报头的一部分，处理部件使偏移标志指向要剥离部分后的第一字节。通信引擎将删除要剥离的部分，将数据重新与字边界对齐，更新 HAF 中的长度字段，并使偏移

字段为零。图 7 中显示了这种情况。此过程的优点是通信引擎要读取的下一状态始终位于 HAF 的确定部分，因此，通信引擎（及处理部件）可配置为访问 HAF 中的同一位置以获得必需的状态信息。而且，可在 HAF 中插入更多具有负偏移值的空间。这种空间插入 HAF 前端。

调度单元 2 可通过将非零值写入标记寄存器发出标记命令。此值将分配给下一输入分组，即，放在报头中。重组单元 3 发出针对此分组的命令时（此时报头完全得到处理），标记值可导致中断产生。对分组进行标记的一个时机是在执行表更新时。可能必需知道在某个时刻前接收的所有分组离开流水线的时间。此类分组需要利用旧表数据进行处理。新的分组要利用新表数据进行处理。由于分组顺序通过流水线后保持不变，因此，这可以通过标记输入分组而实现。在不持分组顺序的处理设备中，可将时间戳添加到每个报头中而不是将一个标记添加到一个报头中。随后，每个报头根据其时间戳进行处理。这涉及在重叠时段存储两种版本的表信息。

每个处理部件可访问多个共享资源，例如，用于诸如查找、管制和统计信息等多种任务的共享资源。此访问是通过与每个处理部件相关联的通信引擎来进行的。提供了多条总线 8a-f，以便将通信引擎连接到共享资源。相同的总线 8a-f 用于传送请求及应答。例如，每个通信引擎 11b 经共享资源总线接口 24b（SRBI-参见图 8）连接到这样的一条总线 8。通信引擎和数据存储器 7b 可经配置总线 21 配置。

通信引擎 11b 最好是处理部件与除其本地存储器 13b 以外的资源进行通信的唯一途径。通信引擎 11b 由主处理部件 14b 经控制接口控制。通信引擎 11b 的主要任务是将分组从一条流水线级传送到下一流水线级。除此之外，它实施上下文切换并与主处理部件 14b 和共享资源通信。

通信引擎 11b 具有连接到流水线的前一电路部件的接收接口 22b

（Rx）和连接到流水线中下一电路部件的发送接口 23b（Tx）。要处理的报头从一个处理单元经通信引擎和 TX 与 RX 接口 22b、23b 传送到另一处理单元。如果不会在特定的处理单元中处理报头，则可以为它配备隧道字段，此字段定义要忽略的处理单元数量。

同时进行接收和发送的通信引擎 11b 的每个发送/接收接口 22b、23b 只可以在不到 50%的时钟周期内访问数据存储器 7。这意味着两个处理级之间的有效带宽小于总线带宽的一半。只要流水线数量大于 2，则这是足够的。然而，第一流水线级必须能够在新分组报头进入流水线时以全总线速度接收突发。同样地，最后的流水线级必须能够以全总线速度产生分组。入口分组缓冲区 4a、5a、6a 负责均衡这些突发。入口分组缓冲区以总线速度接收一个分组报头，然后以其自己的速度将它发送到第一个处理器级。在此期间，不能接收新分组报头。出口分组缓冲区 4f、5f、6f 从最后的处理器级接收分组报头。接收后，它以总线速度将报头发送到分组重组单元 3。入口分组缓冲区还可以具有两个额外任务:

- 添加分组开销。
- 将接收的分组报头中的接口类型/协议代码转换成指向第一任务的指针。分组“层 2”封装包含协议字段，识别“层 3”协议。然而，此字段的意义取决于“层 2”协议。协议字段对（“层 2”协议、“层 3”协议）需要转换成指针，该指针指向要对该分组执行的第一任务。

出口分组缓冲区还有一个额外任务:

- 删除分组开销（的一部分）。

根据本发明，包括了多个硬件扩展以协助 FIFO 管理。

- FIFO 地址偏移。知道当前正在处理的报头的 FIFO 位置后，处理部件可以修改读写地址，使分组看似位于固定地址。
- 自动报头选择。在收到处理引擎的简单请求后，由专用硬件选择可处理的报头。

- 通信引擎选择了新报头时，处理单元可以使用单次读访问获得必需的信息。此信息要分割成不同的目标寄存器（FIFO 位置、报头长度、协议等）。

如上所述，在本发明的一个方面中，可提供诸如通信引擎的硬件，以支持很简单的多任务方案。“上下文切换”在例如处理单元上运行的过程必须等待共享资源的应答，或者报头准备传递到下一级时完成。硬件负责基于 HAF 选择准备要处理的报头。分组经简单的就绪/可用协议或任何其它适当协议从一级传送到另一级。只传送包含相关数据的缓冲区部分。为此，将报头修改为包含引导报头处理所必需的信息。根据本发明的实施例，将分组处理分割成多个任务。每个任务通常处理请求的响应并生成新请求。下一任务的指针存储在报头中。每个任务先计算，然后存储下一任务的指针。每个分组具有一个由不同组合中的两个比特所表示的 Done（完成）和 Ready（就绪）定义的状态。它们具有以下含意:

- Done =0，Ready =0：分组当前正在等待共享资源的响应。无法选择它用于处理，也不可将其发送到下一处理单元的处理部件。
- Done=0，Ready = 1：分组可选择用于在此处理单元上进行处理。
- Done =1，Ready =0：此处理部件上的处理已完成。分组可发送到下一处理单元的处理部件。
- Done =1，Ready =1：未使用

从缓站器管理的角度看，包含分组的缓冲区可处于三种不同的状态下:

- 准备好转到下一级（Ready4Next）
- 准备好被处理（Ready4Processing）
- 等待共享资源应答（Waiting）

通信引擎维护分组状态，例如，通过在寄存器中存储相关状态，

并且还将处于 Ready4Processing 状态下的分组提供给与其相关联的处理器。分组在处理后处于Ready4Next或Waiting状态。处于Ready4Next状态时，通信引擎将分组发送到下一级。处于 Waiting 状态时，该状态将在共享资源应答到达时由通信引擎自动变为 Ready4Processing 或 Ready4Next 状态。

通信引擎用于选择新分组报头。新分组报头的选择由处理部件触发，例如，由系统总线上的处理器读操作触发。当前缓冲区指针保持在寄存器中，指示正由处理部件处理的当前分组。

图 8 显示了根据本发明一个实施例的通信引擎示意图。通信引擎的 5 个主要任务可概述如下：

缓冲区管理：

1）接收端 22（Rx）：从前一处理节点接收分组并推送到双端口 RAM 7 上

2）发送端 23（Tx）：从双端口 RAM 7 弹出就绪分组并发送到下一单元。

多任务（上下文切换））

3）基于缓冲区状态选择适合处理的新分组

共享资源访问：

4）发送端 24a（Tx）：根据请求标识列表装配共享资源请求

5）接收端 24b（Rx）：处理返回共享资源请求的应答。

上述 5 个功能在图 8 中已表示为 4 个有限态机（FSM32、33、34a、34b）和缓冲区管理器 28。应理解，这是通信引擎块的功能描述并且不一定与实际的物理单元相关。图 8 中所示通信引擎的有限态机表示可通过标准处理技术在硬件块中实现。例如，该表示可转换成诸如 Verilog 或 VHDL 等硬件描述语言，并且随后可从 VHDL 源代码自动生成硬件块的网表如门阵列。

由通信引擎处理的主数据结构（列在涉及最多的任务后）为：

- 缓冲区管理：双端口 RAM 的缓冲区中类 FIFO 的数据结构

- 接收报头：写指针寄存器中存储的写指针

- 发送报头：读指针寄存器中存储的读指针

- 多任务：带有下列状态：空（空）、传送就绪（Ready for transfer）、处理就绪（Ready for processing）、传送就绪挂起（Ready for tranfer pending）、处理就绪挂起（Ready for processing pending）加等待级（等待级）之一的缓冲状态矢量，所有状态均存储在缓冲区状态寄存器中，

当前缓冲寄存器中的当前缓冲区，

新分组寄存器：准备要由处理器处理的下一分组的 HAF 和缓冲区位置。

- SR（共享资源）访问：在处理期间，请求在分组缓冲区 RAM 中排队

- 发送端（23a）：维护共享资源请求 FIFO，即在装配请求时允许其它处理的缓冲区

通信引擎的其它部分为：

- RAM 的仲裁器 25：通信引擎的许多功能单元共享到 RAM 7 的总线 19

- 配置接口 26 和 RAM 7 中通信引擎和缓冲区的配置字段图。控制接口 26 可用于配置通信引擎，例如，寄存器和随机存取存储器大小。

数据存储器 7 的一个端口经数据存储器（DM）RAM 接口 27 和总线 19 连接到通信引擎 11。在正常操作期间，此总线 19 用于以到达通信引擎 11 的 RX 接口 22 的数据填充存储器 7 中的分组缓冲区 37a-h，或者将它清空到 TX 接口 23，这两种操作均通过 RAM 仲裁器 25 进行。仲裁器 25 对功能单元（FSM）：SR RX 34b、SR TX 34a、下一分组选择 29、接收 32、发送 33 之间到 DM RAM 7 的访问加以

组织并排定优先处理顺序。

每个处理器单元 14 可访问用于查找、管制和统计信息的多个共享资源。提供了多条总线 8，以将处理单元 14 连接到共享资源。同样的总线 8 可用于传送请求及应答。每个通信引擎 11b 经共享资源总线接口 24b（SRBI）连接到此类总线。

每个通信引擎 11 维护多个分组缓冲区 37a-h。每个缓冲区可以包含一个分组，即，具有存储一个分组的装置。针对分组接收与发送，将缓冲区作为 FIFO 处理，因此，分组顺序保持不变。分组从 RX 接口 22 进入，而通过 TX 接口 23 离开。数据存储器 7 中缓冲区的数量、缓冲区大小和缓冲区的起始点通过控制接口 26 配置。缓冲区大小始终是 2 的幂，并且缓冲区起始点始终为缓冲区大小的倍数。这样，每个存储器地址可容易地分解成缓冲区编号和缓冲区中的偏移。每个缓冲区可包含一个分组的数据。处理部件 14 对缓冲区的写访问由通信引擎 11 经监控总线 18 监控，并相应地在缓冲区状态寄存器中更新缓冲区状态。缓冲区管理器 28 在寄存器 35 中维护 4 个指针，其中两个指针指向缓冲区，而另外两个指向缓冲区中的特定字：

- 接收写指针：指向接收数据时将写入的下一字。复位后，它指向第一缓冲区的第一字。
- 发送写指针：指向发送数据时将要读取的下一字。在复位后，它指向第一缓冲区的第一个字。
- 最后发送缓冲区指针：指向最后发送的缓冲区，或者指向正在发送的缓冲区，即一读取某个缓冲区的第一字，便将其更新以指向该缓冲区。复位后，它指向最后的缓冲区。
- 当前缓冲区指针：指向当前处理器在使用的缓冲区。相关联的当前缓冲区有效标志指示当前缓冲区的内容是否有效。处理单元不在处理任何分组时，当前缓冲区有效会被清除。

图 9 示意性地显示了各种指针。

在缓冲区状态寄存器 30 中维护对应各缓冲区的状态。每个缓冲

区处于以下 5 种状态之一:

- 空: 缓冲区不包含分组。
- 传送就绪: 缓冲区中的分组可传送到下一处理器级。
- 处理就绪: 缓冲区中的分组可由处理器选择进行处理。
- ReadyForTransferWSRPending: 当所有共享资源请求均被发送了时，分组必须转到传送就绪状态。
- ReadyForProcessingWSRPending: 当所有共享资源请求均被发送了时，分组必须转到传送就绪状态。

除状态外，在寄存器 35 中为每个缓冲区维护一个等待级。等待级不等于零表示分组在等待某一事件，既不应移交处理器，也不应发送。通常，等待级表示正在进行的共享资源请求的数量。复位后，所有缓冲区均处于空状态。分组完整接收后，对需要加以处理的分组，将存储分组的缓冲区状态更新为处理就绪状态，而对于不需要处理的分组（例如，丢弃的分组），该缓冲区则更新为传送就绪状态。对于任一输入分组，缓冲区的等待级设为零。

处理分组后，处理器 14 通过将传送和共享资源请求比特写入 HAF，即双端口 RAM 7 的相关缓冲区中，从而更新该分组的缓冲区状态。此写操作由通信引擎 11 经监控总线 18 进行监控。处理器 14 可以在没有共享资源请求要发送时将缓冲区置于处理就绪或传送就绪状态，或者在有请求要发送时置于 ReadyForTransferWSRPending 或 ReadyForProcessingWSRPending 状态。所有请求一发送，缓冲区状态就立刻从 ReadyForTransferWSRPending 或 ReadyForProcessingWSRPending 状态返回传送就绪或处理就绪状态。读指针到达新缓冲区的开始处时，它会在读取和发送分组前等待，直至该缓冲区进入传送就绪状态且等待级等于零。发送一开始，缓冲区状态立即设为空。这保证无法再选择分组（因为写指针决不会越过读指针，因此，即便缓冲区处于空状态，也不会覆盖未发送数据）。

只要存在空缓冲区，便可从 RX 接口接收输入数据。在写指针到达读指针时，缓冲区为满（由于在两种状态下读指针均等于写指针，因此需要额外的标记区分全满与空）。

当读指针指向缓冲区，而该缓冲区进入传送就绪状态且等待级为零时会触发分组发送。首先，缓冲区状态设为空，随后，从 RAM 读取并发送 HAF 和临时暂存区。只包含要剥离开销的字被忽略。随后，在发送前读取并重新对齐分组数据的其余部分，以删除第一字中的剩余开销字节。然而，如果分组设置了其丢弃标志，则不读取该分组数据。分组发送后，读指针跳到下一缓冲区的开始处。

通信引擎维护当前缓冲区指针，它指向处理部件当前处理的分组缓冲区。相关联的有效标志指示当前缓冲区的内容有效。如果处理器不在处理分组，则有效标志设为假。有 5 种不同的算法可用于选择新分组：

- 第一分组(0)：返回包含最老分组的缓冲区。
- 下一分组(1)：返回包含分组的当前缓冲区之后的第一缓冲区。如果无当前缓冲区，则如第一分组算法那样工作。
- 第一可处理分组(2)：返回包含处于处理就绪状态下的最老分组的缓冲区。
- 下一可处理分组(3)：返回包含处于处理就绪状态下的分组的当前缓冲区之后的第一缓冲区。如果无当前缓冲区，则象第一可处理分组算法一样工作。
- 下一缓冲区(4)：返回当前缓冲区之后的第一缓冲区。如果无当前缓冲区，则返回第一缓冲区。

处理器完成对一个缓冲区的处理时，它指定要对该分组执行的下一任务。这是通过在分组 HAF 中写入以下字段来完成的：

- 任务：指向下一任务的指针。
- 隧道：如果下一任务不在此处理器或下一处理器上，则设置此字段。

- 丢弃：如果分组需要被丢弃则设置此字段。使任务和隧道字段无效。
- 传送：如果下一任务在另一处理器上，则设置此字段；而如果下一任务在同一处理器上则将此字段清零。
- 共享资源请求：如果在切换到下一任务前必须完成共享资源访问，则设置此字段。

传送和共享资源请求比特不仅写入存储器，还由通信引擎经XLMI接口监控。这用于更新缓冲区状态：

- 共享资源请求=0，并且传送=0：处理就绪
- 共享资源请求=0，并且传送=1：传送就绪
- 共享资源请求=1，并且传送=0：ReadyForProcessingWSRPending
- 共享资源请求=1，并且传送=1：ReadyForTransferWSRPending

通信引擎11提供到共享资源的通用接口24。请求由标题及其后发送到共享资源的的数据块组成。通信引擎11在SRTX 34a中生成标题，但数据必须由处理器14提供。根据要发送的数据大小和性质，可以区分三种装配请求的方式：

- 即时：要发送的数据是请求标识的一部分。这适用于只包含少量数据的请求。针对请求的应答存储在请求标识中偏移字段指示的位置（偏移）或默认偏移（默认）。
- 存储器：要发送的数据存储在存储器中。请求标识包含数据的位置和大小。提供了两种请求类型：在一种类型中，数据位于分组缓冲区中（相对）；而在另一种类型中，位置指向绝对存储器地址（绝对）。偏移字段指示应答必须存储在缓冲区中的位置。
- 定序器：小的定序器从所有分组收集数据，并构建请求。请求标识包含指向定序器程序开始的指针。偏移字段指示应答必须存储在缓冲区中的位置。

共享资源请求标识可包含以下字段：

- 请求类型：如上所述确定请求的类型。

- 资源：要寻址的资源标识
- 成功比特：要使用的成功比特的索引（参见下文）
- 命令：如果置位，则表示无需对此请求作出应答。如果清零，期望有应答。
- 最后标志：为分组的最后请求标识而设。对于其它请求标识则清零。
- 偏移：缓冲区中请求的应答必须存放的位置。偏移以字节为单位，从缓冲区的起始处开始。
- 结尾偏移（EndOffset）：如果设置，则指示应答的结尾必须定位的位置。偏移于是指向应答后的第一字节。如果清零，则偏移指向应答的第一字节必须存放的位置。
- 数据：请求中要发送的数据，用于即时请求。
- 地址：要发送的数据所处的位置（绝对地址或相对于分组缓冲区开始处的相对地址），用于存储器请求。
- 长度：要发送的字数量，用于存储器请求。
- 程序：定序器要执行的程序的开始地址。

在将请求标识置于缓冲存储器 7 中后，处理器通过将 HAF 中的共享资源请求比特置位，指示存在这些标识（这通常在为下一任务更新 HAF 时完成）。

处理器释放缓冲区（通过请求新的缓冲区）时会检查 HAF 中的共享资源请求比特。这可以通过评估缓冲区状态来完成。如果已设置，则将此分组的缓冲区编号压入小 FIFO，即共享资源请求 FIFO 中。当此 FIFO 为满时，在有新分组请求时会返回空闲任务，以避免溢出。SR TX 状态机 34a（图 8）从共享资源请求 FIFO 弹出缓冲区编号。它随后从最高地址开始解析缓冲区中的请求标识，直到遇到其 Last 比特被置位的请求标识。随后，从 FIFO 中弹出下一缓冲区编号，直至再无条目可用为止。每次解析请求标识时，会将对应的请求放在一起并发送到 SRBI 总线 24a。在设置了 HAF 的共享资源请求

比特时，根据传送比特的值，将对应的缓冲区状态设为ReadyForTransferWSRPending 或 ReadyForProcessingWSRPending。只要缓冲区处于这些状态之一，它就不适于发送或处理。

无论何时发送非命令请求，等待级字段值会加 1。收到应答时，它会减 1。所有请求发送后，缓冲区状态设为传送就绪（当来自 ReadyForTransferWSRPending 状态时）或处理就绪（当来自 ReadyForProcessingWSRPending 状态时）。此机制保证只可在不早于如下时刻发送或处理分组（采用下一可处理分组算法/第一可处理分组算法）：

- 所有请求均已发送
- 已发送请求的所有应答均已到达。

应答的目的地址通过共享资源总线套接字（bus socket）解码。匹配本地地址的应答由通信引擎经 SRBI RX 接口 24b 接收。应答标题包含必须存储应答的缓冲区编号和偏移。基于此，通信引擎能够计算绝对存储器地址。应答的数据部分从 SRBI 总线 8 接收，并存储到数据存储器 7 中。当所有数据均已存储时，可通过对可寻址缓冲区中的 HAF 执行读取－修改－写入操作，从而更新成功比特（参见下文），最后，使该缓冲区的等待级字段值减 1。

一些共享资源请求可以成功或失败状态结束（例如，精确匹配资源将一个地址与地址列表进行比较。匹配返回标识符，不匹配返回失败状态）。添加装置，以将此传播到所涉及分组的 HAF。在 HAF 中设置了多个比特，例如 5 个比特，用于捕获不同请求的结果。因此，请求标识必需指定要使用所述 5 个比特中的哪一个比特。共享资源还可以置于链中，即第一共享资源的结果是第二共享资源的请求，并以此类推。这些共享资源中的每个共享资源可具有成功或失败状态，因此可能需要其自己的成功比特。重要的是要注意，在资源以失败状态终止时，请求链会中断。这种情况下，失败的资源直接将其应答发送到始发通信引擎。

在处理分组时，与通信引擎 11 相关联的处理部件 14 可以通过将必需的请求标识写入相关分组的缓冲区中而使通信引擎 11 发出一个或多个到共享资源的请求。例如，每个请求标识为例如单个 64 比特字，并将促使生成一个共享资源请求。装配请求并将其发送到共享资源的过程最好在处理器不再处理分组时开始。分组只在从共享资源传来的所有应答到达后才可以再次选择以加以处理。这保证了处理器和通信引擎决不会同时修改单个缓冲区。

共享资源请求通过将请求信息连同下一操作的信息从处理部件发送到相关联的通信引擎来调用。下一操作信息是识别需要在此分组上执行的下一操作的指针以及指示需要为该操作将分组传送到下一处理单元的选项。接着，处理单元读取随后需要执行的操作的指针。此选择由同一专用硬件如通信引擎完成，该专用硬件为与处理单元相关的处理部件控制将报头拷贝入或拷贝出缓冲存储器 7。就此而言，通信引擎还处理来自共享资源的应答。共享资源请求最好包括对发出请求的处理部件的引用。当应答从共享资源返回时，应答包括此引用。这允许接收通信引擎将应答写入正确位置中，即写入其缓冲区中的相关报头中。随后，处理部件跳到识别出的操作。这样，处理模型为执行的单线程模型。不需要需要保存所有处理部件状态的高代价上下文切换，即不需要可能在时间或硬件上代价高的操作。另外，它减少了选择此类处理部件的选项数量。执行的单线程模型实际上是如下所示的无限循环:

1. 读操作信息
2. 跳到该操作
3. 将对共享资源的请求格式化或者指示将分组移交下一级
4. 返回步骤 1。

此编程模型如此严格定义了将对单个分组执行的后续操作以及将执行这些操作的级。它未定义在单个处理部件上执行的（操作，分组）元组的顺序。该顺序是由共享资源定时和等待时间决定的，

并且这样的确切行为对该编程模型是透明的。

此编程模型的严格定义允许在不需要包括这些定时和等待时间数字的细节的层次上验证要对分组执行的操作的程序代码。

本发明的又一实施例涉及如何访问共享资源。处理单元和共享资源经多条总线连接，例如，经双 64 位宽总线连接。每个节点（假设为处理单元或共享资源）具有到这些总线中的一条或多条总线的连接。总线的数量和连接到每条总线的节点数量由带宽要求确定。每个节点最好锁定总线，以避免长连接。这可实现高速度，但同时带来较高的等待时间。所有节点具有相同的优先级，并且仲裁是在每个节点中以分布方式完成的。只要在总线上检测到分组结尾，每个节点便可插入分组。插入分组时，使输入业务停止。假定实际带宽不太接近于可用带宽时这种简单仲裁就已足够且等待时间不太重要。后者对分组处理器而言是正确的，而前者可通过适当选择总线拓朴来实现。

共享资源可连接到如图 10 所示的 2 位宽总线。处理单元 P1 到 P8 排列在一条总线上，并可以访问共享资源 SR1 和 SR2；处理单元 P9 到 P16 排列在第二条总线上，且只可以访问 SR2；处理单元 P17、P19、P21、P23 到 P24 排列在第三条总线上，且只可以访问 SR3；以及处理单元 P18、P20、P22、P24 排列在第四条总线上，且只可以访问 SR3。处理节点通过向共享总线上其它节点相互发送消息而与共享资源进行通信。总线上的每个节点具有唯一地址。每当总线闲置，每个节点就可以在总线上插入分组。分组的目的节点从总线取走分组。总线上提供了争用机制以防止冲突。每个沿总线传递的请求由相关的共享资源选择并加以处理，然后再将响应放置到总线上。

总线可采用环状形式的总线而非图 10 所示的总线类型，并且响应可沿环路遍历，直至到达相关处理单元/共享资源，这时，由该处理单元/共享资源接收。

本领域的技术人员由上述内容可以理解，进入处理流水线 4-6

的分组会触发一系列在该分组的处理流水线上执行的操作。操作被定义为一小段程序（假定在硬件或软件中）代码，这段代码在一些时钟周期期间在处理部件上执行，同时不与任何共享资源交互或者不与流水线中的下一处理部件通信。一个操作在收到共享资源的请求时结束，或者通过将分组移交到下一级而结束。图 6 以流程图的形式示意性地说明了这种操作、共享资源请求和显示分组移交的序列。先从调度单元传来分组报头。流水线第一级处理单元的处理部件对该报头执行操作。随后，请求共享资源 SR1。在等待应答的时间内，报头保持在相关联的 FIFO 存储器中。在收到应答时，由同一处理部件执行第二操作。因此，在一个处理部件内，可以对同一分组执行这些操作中的几个操作。在一个处理部件对一个报头的处理结束时，将修改过的报头传送到下一级，以对其执行进一步的操作。

图 11 示意性地显示了流水线中处理单元对分组进行的处理的流程图。如前所述，在缓冲存储器 7 内，每个缓冲区可能处于下列可能的缓冲区状态之一中：

空

R4P：处理就绪

R4T：传送就绪

R4PwSRPending：共享资源请求发送后处理就绪

R4TwSRPending：共享资源请求发送后传送就绪

等待级：未决共享资源请求数量

HAF 中的相关比特：

传送

共享资源请求

在步骤 100 中，新分组报头出现在通信引擎的接收端口，如果存储器中存在可用（空）缓冲区位置，则接收分组报头，可用缓冲

区的状态通过缓冲区管理器加以访问。如果存在可用缓冲区，则报头数据在步骤 102 中发送到存储器，并在步骤 104 中存储在适当的缓冲区中，即存储在适当的存储器位置上。在步骤 106 中，如果要对报头进行处理，则由通信引擎将缓冲区状态寄存器中的缓冲区状态从空状态更新为 R4P（或者，对于不需要处理的分组报头，例如要丢弃和进行隧道处理的分组报头，更新为 R4T）。随着缓冲区中较老的分组报头得到处理并进一步沿流水线向下发送，经过一段时间，目前的 R4P 分组报头就可供选择了。

在步骤 108 中，处理部件结束对前一分组报头的处理，并向通信引擎请求下一分组报头。下一分组选择是在步骤 110 中根据缓冲区状态寄存器中包含的缓冲区状态来决定的。如果没有 R4P 分组报头可用，则通信引擎向处理器返回空闲应答。处理部件将再次进行相同的请求，直至得到非空闲应答。

在步骤 114 中，通信引擎访问下一分组寄存器，并发送下一分组报头位置，并且将相关联的任务指针也发送给处理部件。为让处理部件立即启动，不仅在应答中提供了下一分组报头位置，而且还给出了相关联的任务指针。此数据是要处理的下一分组报头的 HAF 的一部分，因而需要对存储器的读周期。因此，通信引擎在步骤 112 中用分组报头位置加任务指针二元组连续更新新分组寄存器，以使此 HAF 读操作在处理部件周期预算外进行。

在步骤 116 中，处理部件处理分组报头并更新 HAF 字段“传送”和“共享资源请求”。通信引擎监视数据总线并依据处理部件与存储器之间的这种总线监视，在步骤 118 中通知缓冲区状态管理器更新缓冲区状态。例如，如果未发送共享资源请求，则分组报头状态可变为 R4P 或 R4T，如果要发送共享资源请求，则可变为 R4PwSRPending 或 R4TwSRPending。

在步骤 120 中，待决共享资源请求在处理阶段后触发 SR 发送状态机，以便装配并发送列在缓冲区末尾的共享资源请求，即请求标

识列表。在步骤 122 中，按顺序处理请求 ID。间接类型请求需要从存储器读取。在步骤 124 中，对于与命令相反，期望返回应答的每个请求，相应使等待级计数器递增。

在步骤 126 中，SR 接收状态机在收到 SR 应答时处理结果，并在步骤 128 中写入存储器，更具体地说，写入与相应分组报头相关联的缓冲区位置。在步骤 130 中，使等待级计数器递减。

最终，当所有请求均已发送且所有应答均已收到后，在步骤 132 中将分组报头状态设为 R4P 或 R4T。在缓冲区中对分组报头数据流采用了先入先出的处理方法。在步骤 134 中，在当前最老的分组报头状态变为“R4T”时，发送状态机随后会将此分组报头输出到发送端口。

根据本发明的处理流水线满足以下要求:

- 通信开销很低，以满足非常有限的周期预算
- 可以支持不将分组重新排序的选项
- 在分组标题大小保持不变、从中剥离信息或者在其中添加信息的情况下，通过流水线的报头相应地保持相同大小、缩小或者增大；流水线始终使下一相关标题与处理器字边界重新对齐。这使得第一标题看似在 FIFO 存储器 7b...7d 中的固定位置上，这简化了软件。
- 处理单元能够读取、剥离和修改报头；处理单元不感兴趣的项目会传送到下一级而无需处理单元干预。因此，不会损坏标题中承载的有效负荷部分，而只是将其转发。
- 处理单元能够丢弃分组。

处理单元是同步的。

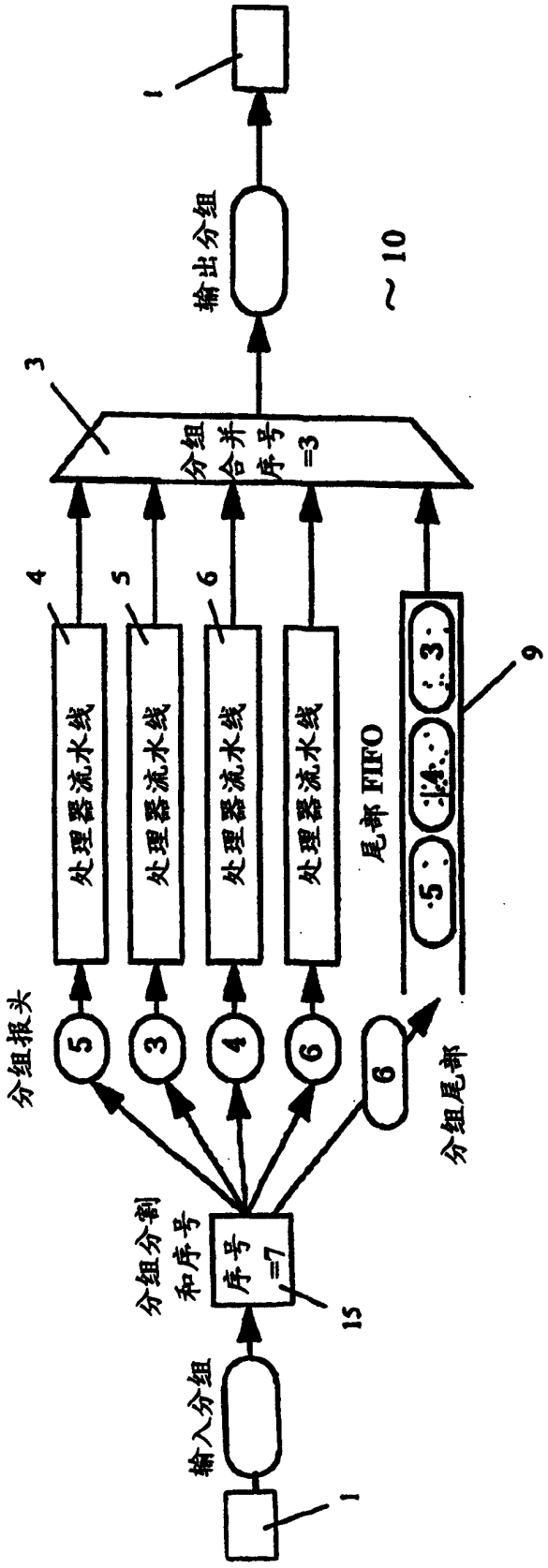

1
输出分组
~10
3
分组合并序号=3
4
5
6
处理器流水线
处理器流水线
处理器流水线
处理器流水线
尾部 FIFO
9
分组报头
5
3
4
6
6
分组尾部
分组分割和序号
序号=7
15
输入分组
1

图 1a

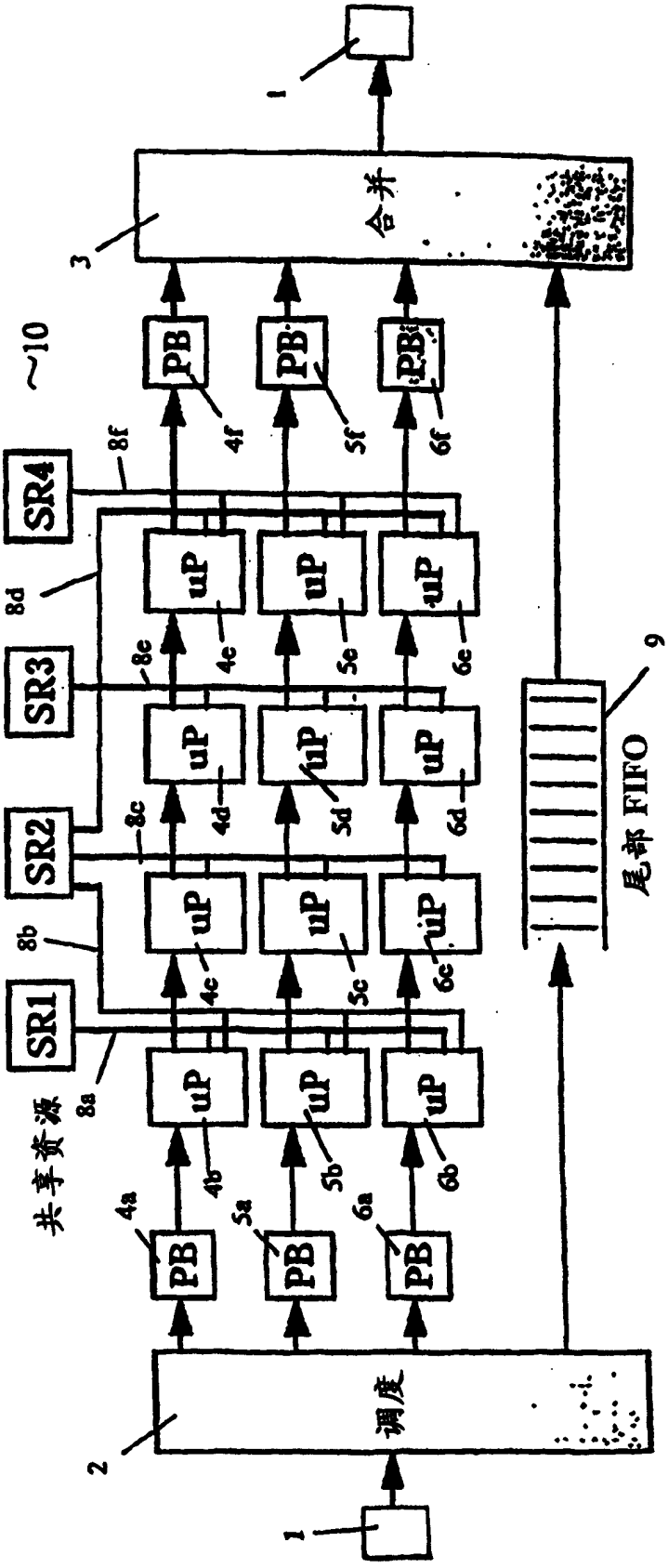

1
合并
3
~10
PB
4f
5f
6f
SR4
8f
8d
uP
4e
5e
6e
SR3
8e
4d
5d
6d
SR2
8c
4c
5c
6c
SR1
8b
8a
共享资源
4b
5b
6b
4a
5a
6a
调度
2
1
尾部 FIFO
9

图 1b

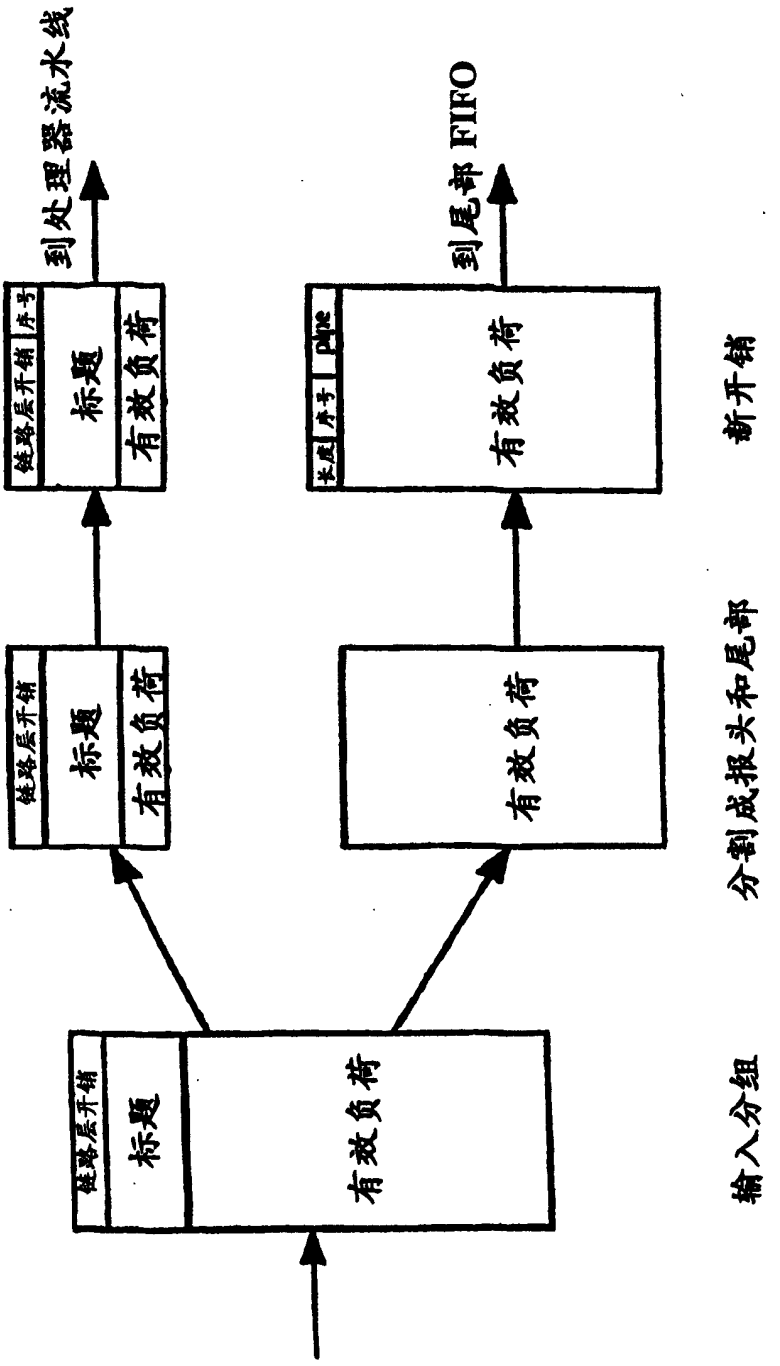


图 2a

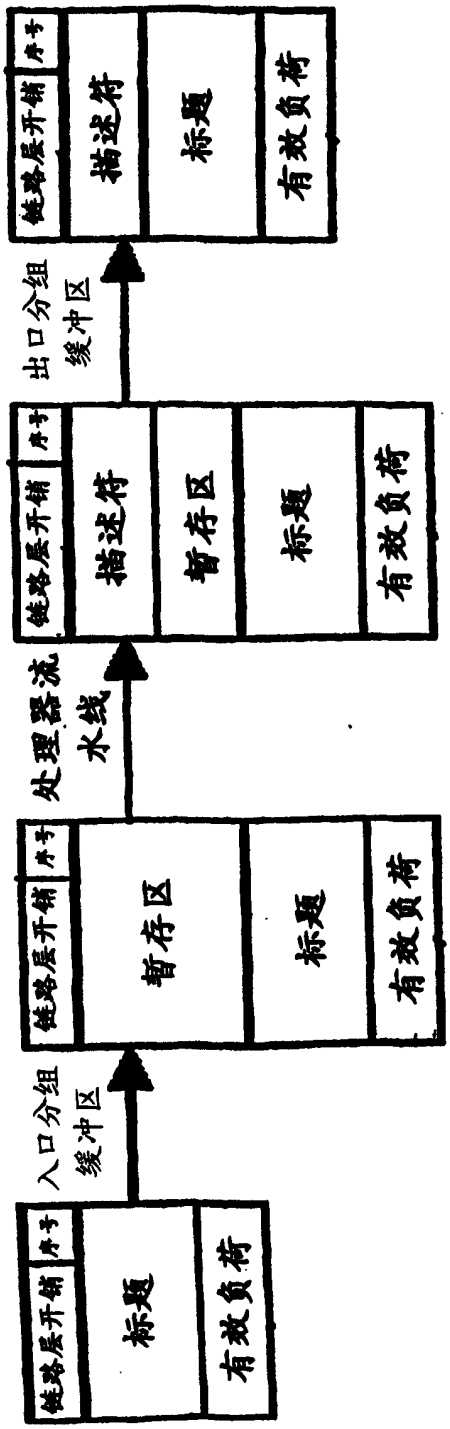

链路层开销
序号
标题
有效负荷
入口分组
缓冲区
链路层开销
序号
暂存区
标题
有效负荷
处理器流
水线
链路层开销
序号
描述符
暂存区
标题
有效负荷
出口分组
缓冲区
链路层开销
序号
描述符
标题
有效负荷

图 2b

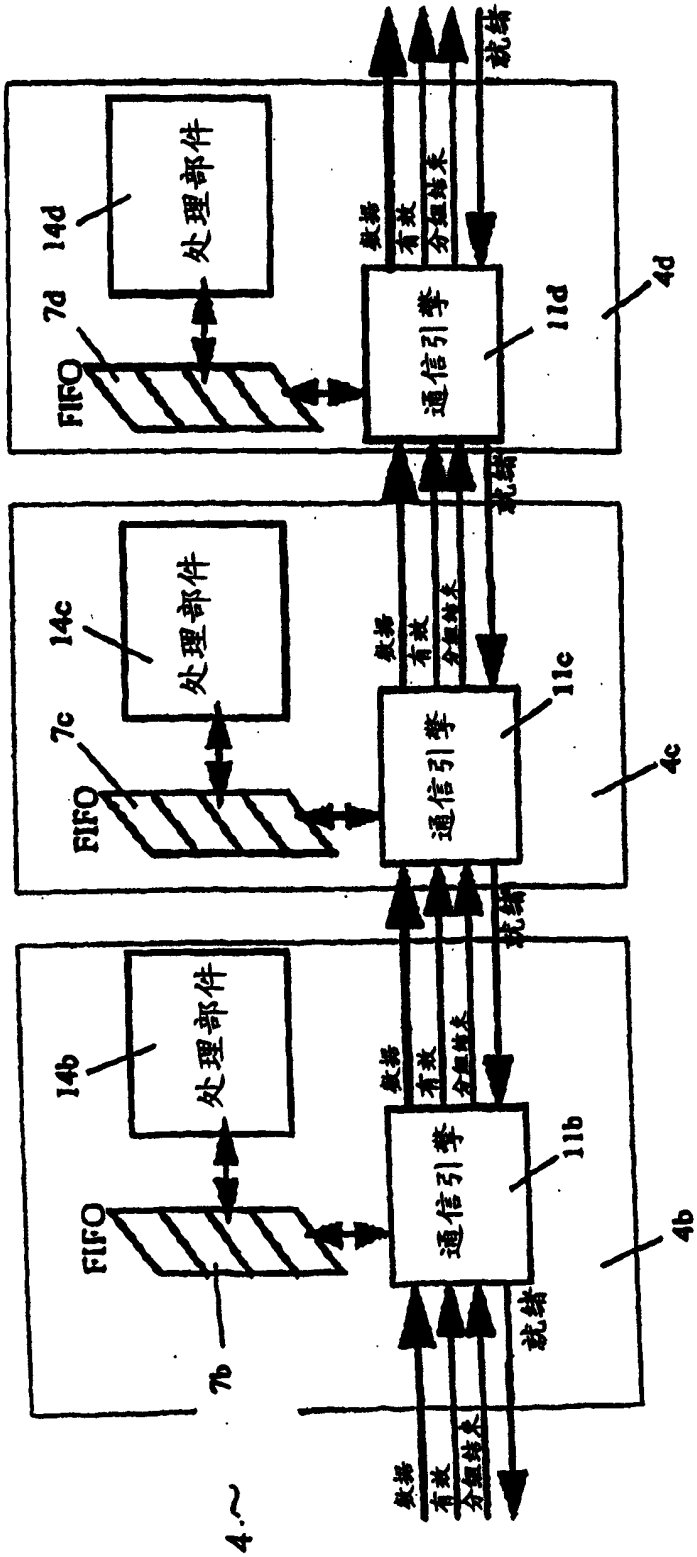

处理部件
14d
7d
FIFO
通信引擎
11d
4d
数据
有效
分组结束
就绪
处理部件
14c
7c
FIFO
通信引擎
11c
4c
处理部件
14b
FIFO
通信引擎
11b
4b
7b
4

图 3

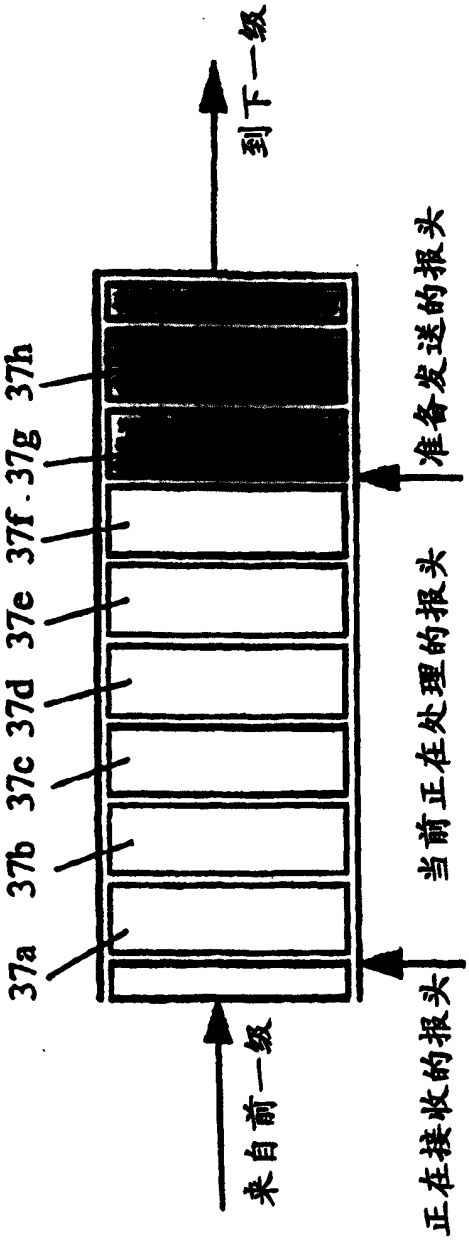

到下一级
准备发送的报头
当前正在处理的报头
正在接收的报头
来自前一级
37a
37b
37c
37d
37e
37f
37g
37h

图 4a

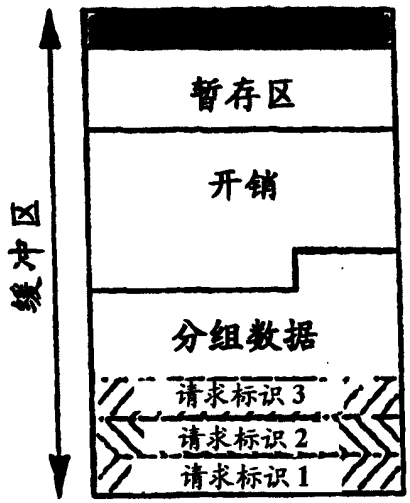


图 4b

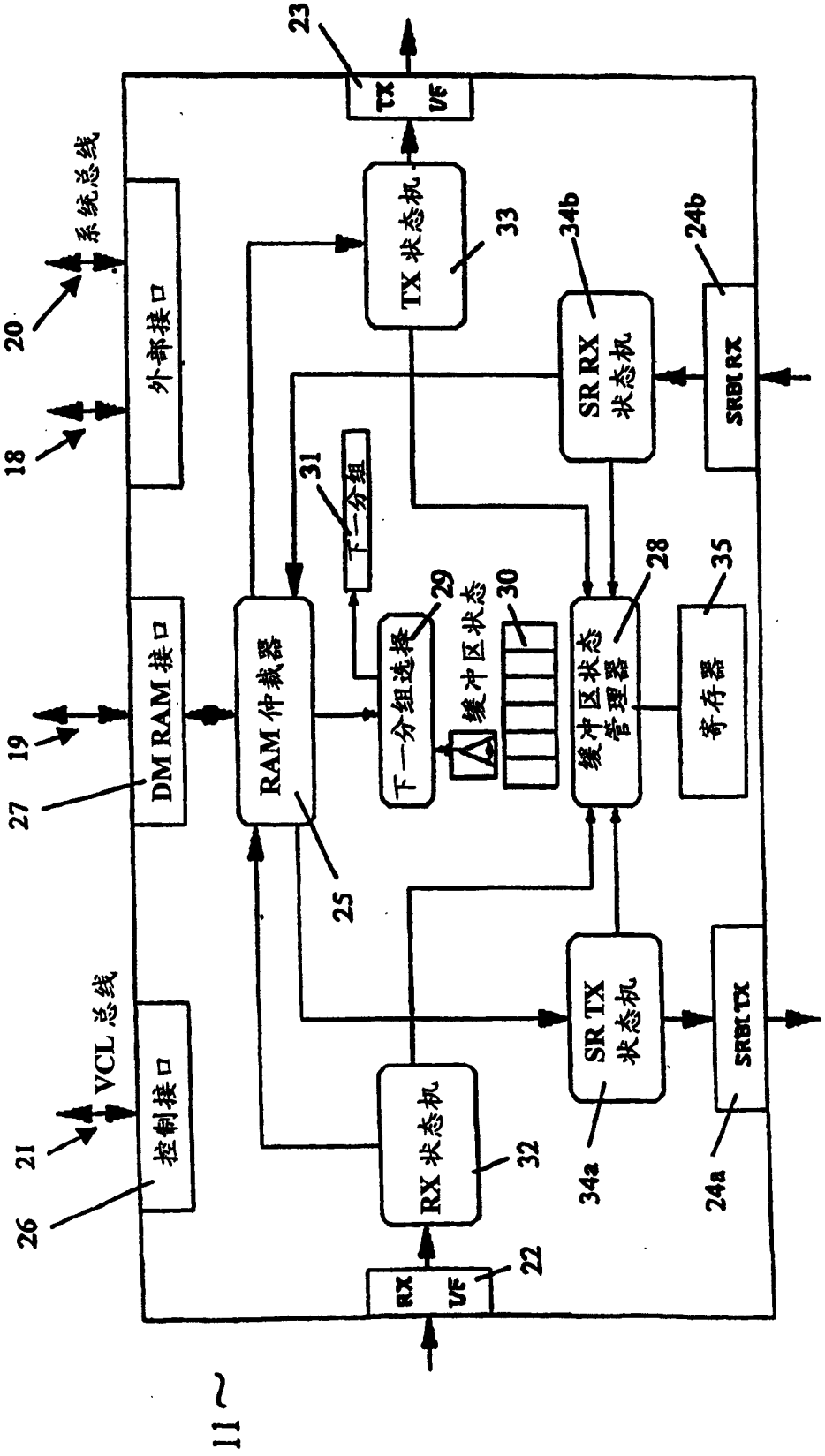

11
外部接口
DM RAM 接口
控制接口
RAM 仲裁器
下一分组选择
下一分组
缓冲区状态
缓冲区状态管理器
寄存器
TX 状态机
RX 状态机
SR TX 状态机
SR RX 状态机
SRBI TX
SRBI RX
TX
RX
VF
系统总线
VCL 总线
18
19
20
21
22
23
24a
24b
25
26
27
28
29
30
31
32
33
34a
34b
35

图 8

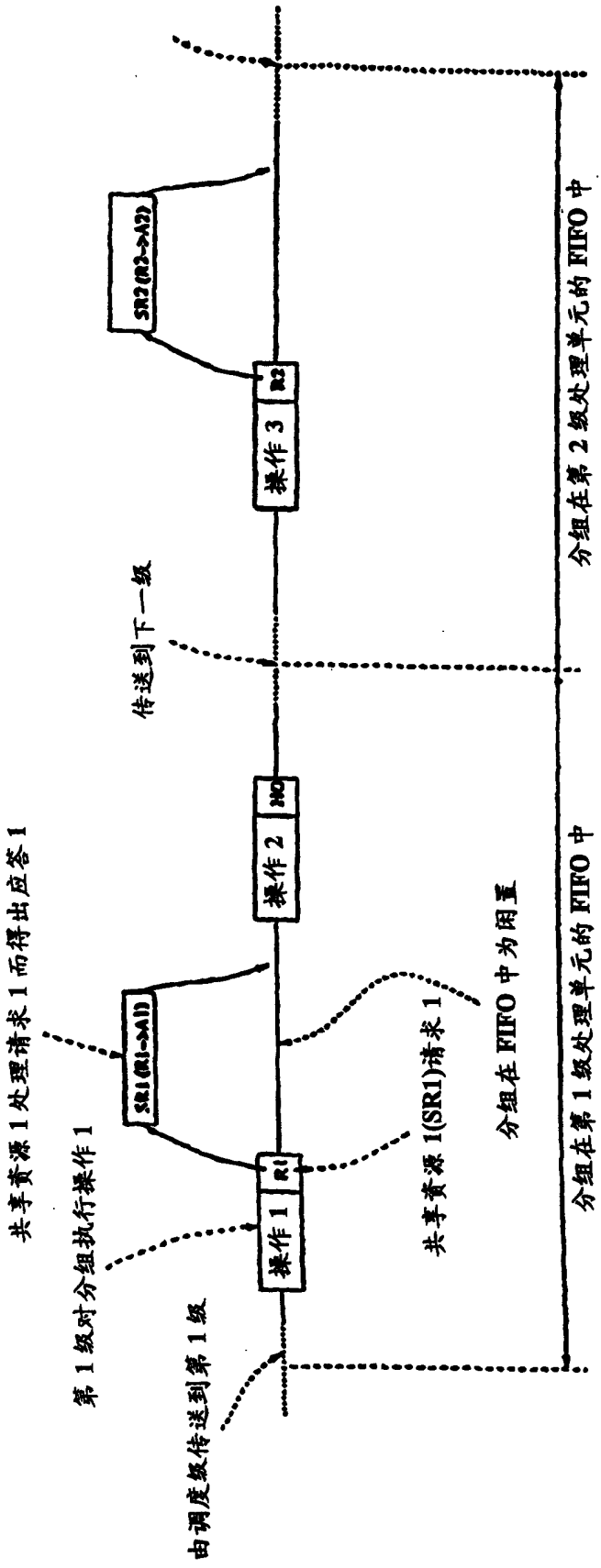


图 6 流水线中对分组如何进行处理

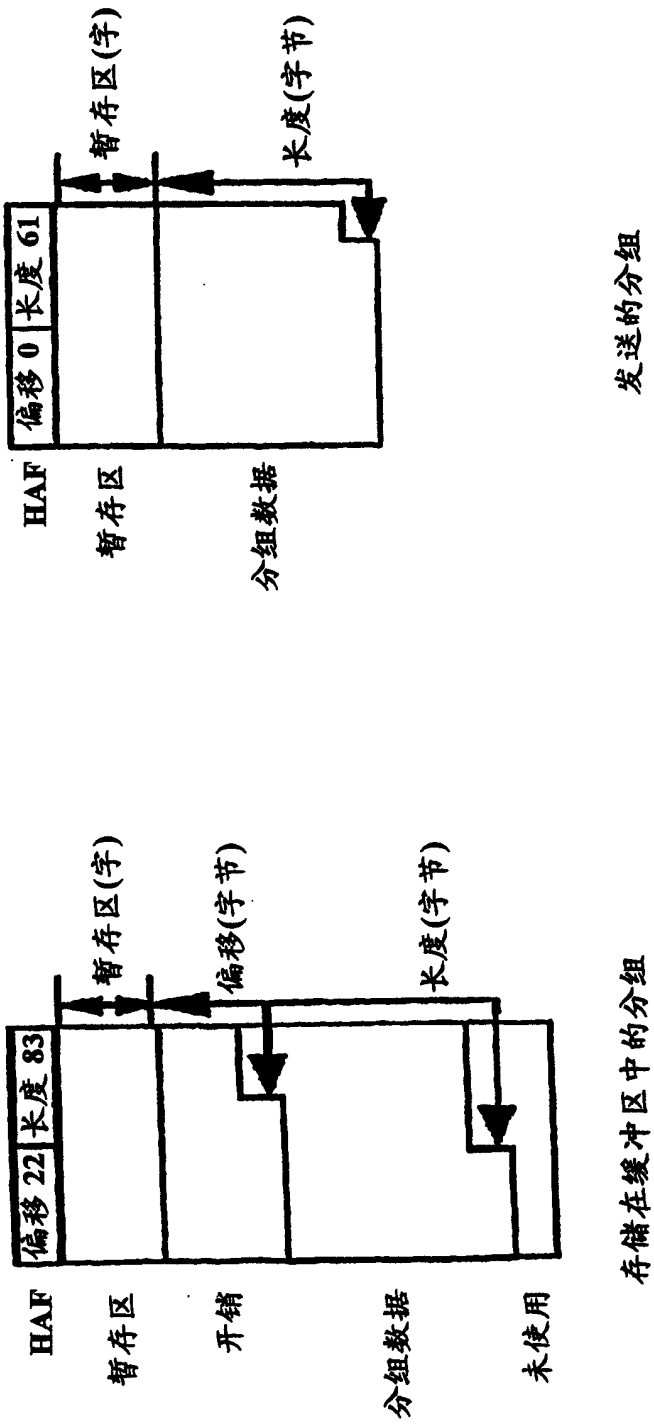

偏移 0
长度 61
暂存区(字)
长度(字节)
HAF
暂存区
分组数据
发送的分组
偏移 22
长度 83
暂存区(字)
偏移(字节)
长度(字节)
HAF
暂存区
开销
分组数据
未使用
存储在缓冲区中的分组

图 7

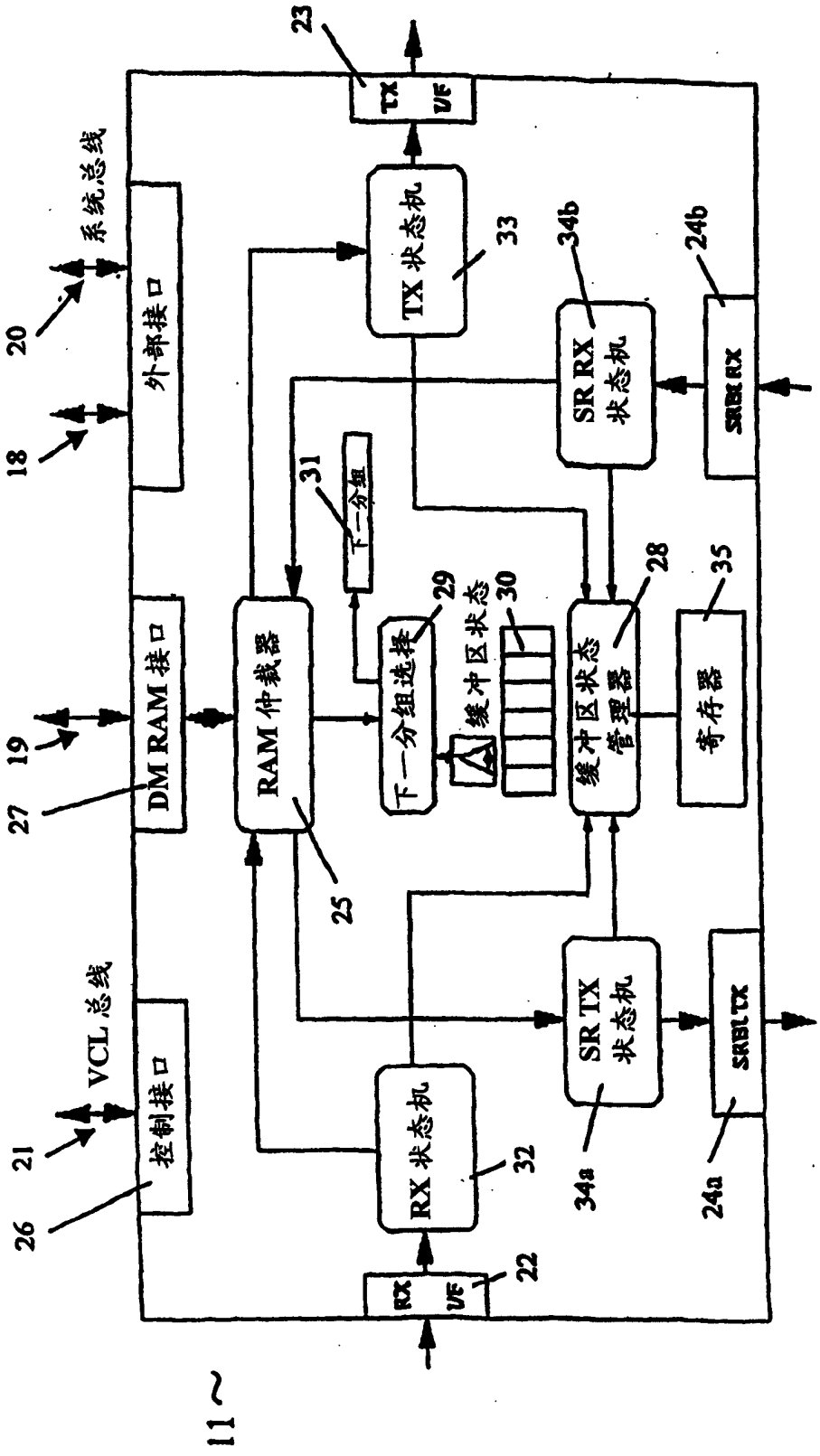

11
TX
VF
23
TX 状态机
33
34b
24b
SR RX 状态机
SRBI RX
系统总线
20
外部接口
18
31
下一分组
29
30
缓冲区状态
28
35
寄存器
19
DM RAM 接口
27
RAM 仲裁器
下一分组选择
25
缓冲区状态管理器
VCL 总线
21
控制接口
26
SR TX 状态机
SRBI TX
RX 状态机
32
34a
24a
RX
VF
22

图 8

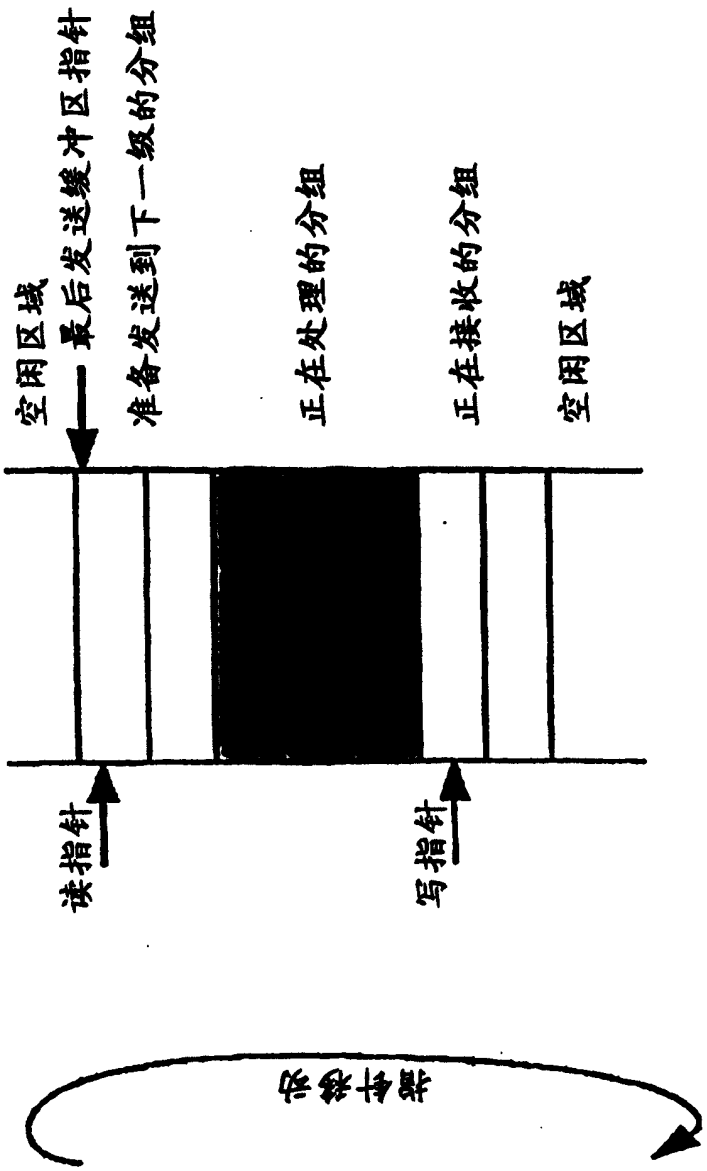

空闲区域
最后发送缓冲区指针
准备发送到下一级的分组
正在处理的分组
正在接收的分组
空闲区域
读指针
写指针
指针移动

图 9

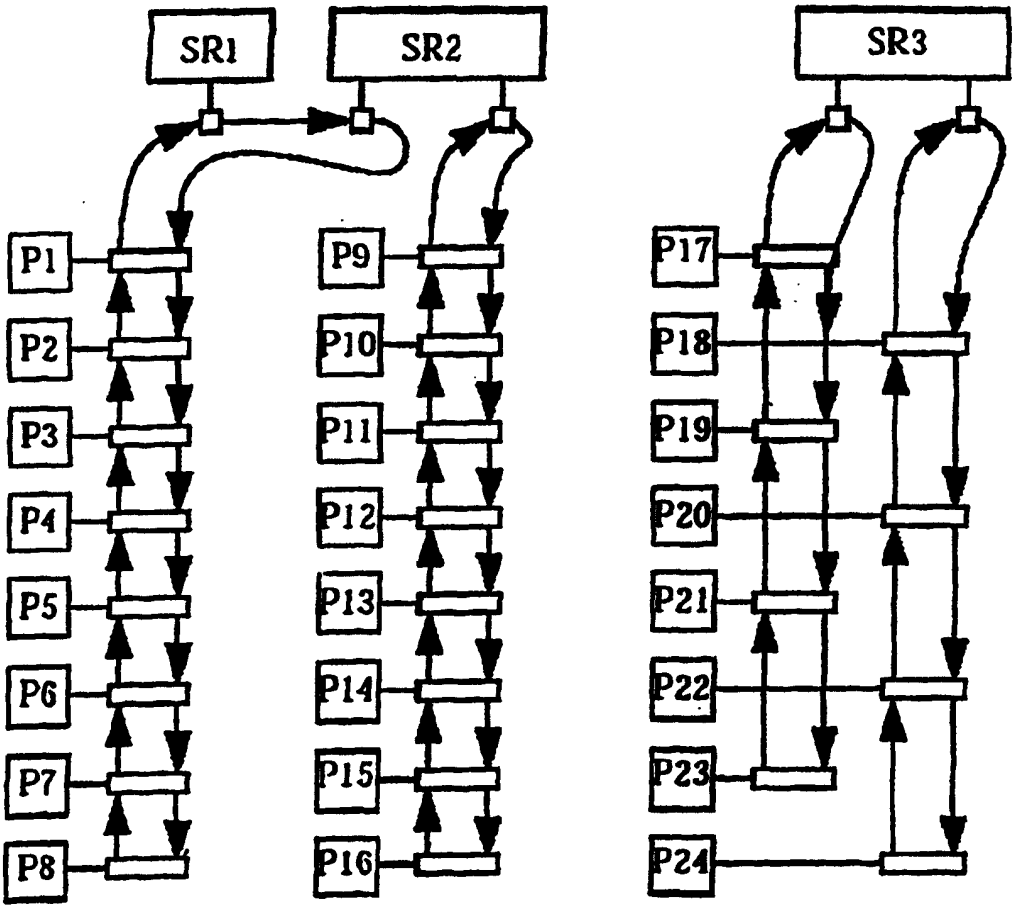

SR1
SR2
SR3
P1
P2
P3
P4
P5
P6
P7
P8
P9
P10
P11
P12
P13
P14
P15
P16
P17
P18
P19
P20
P21
P22
P23
P24

图 10

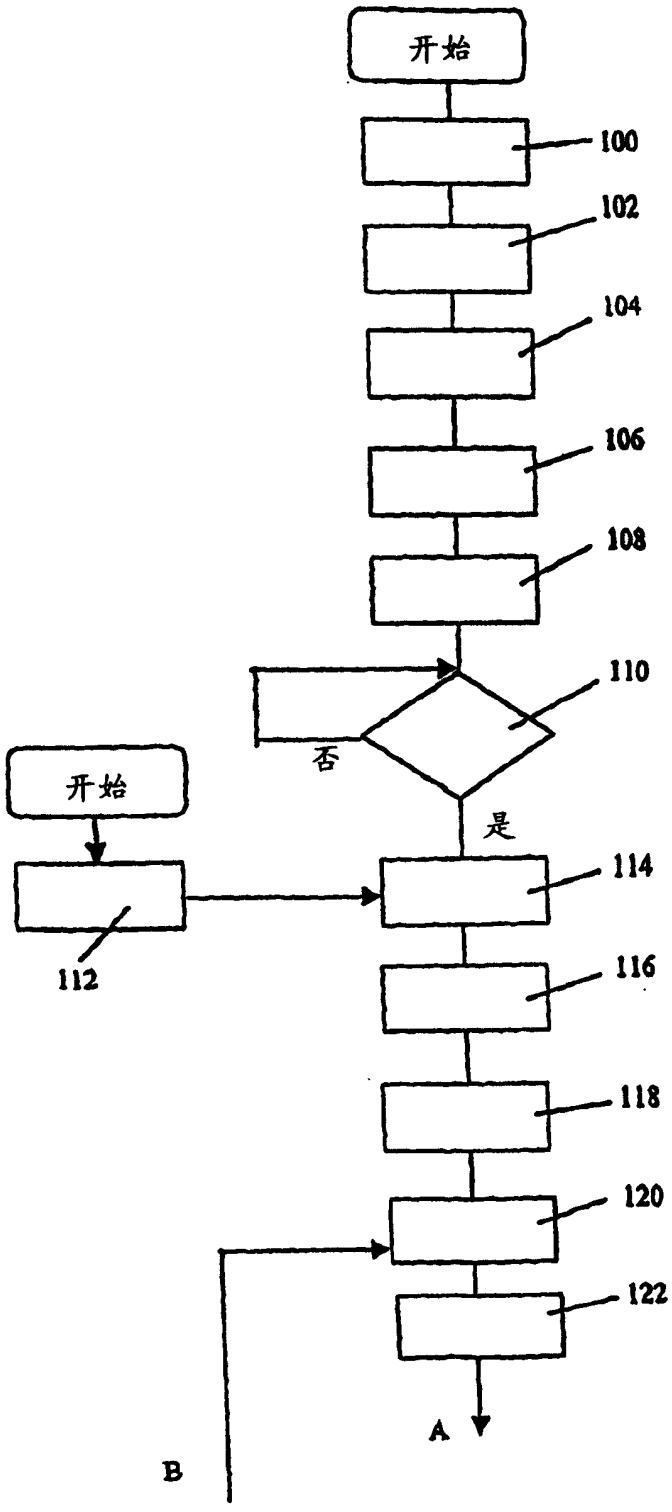


图 11

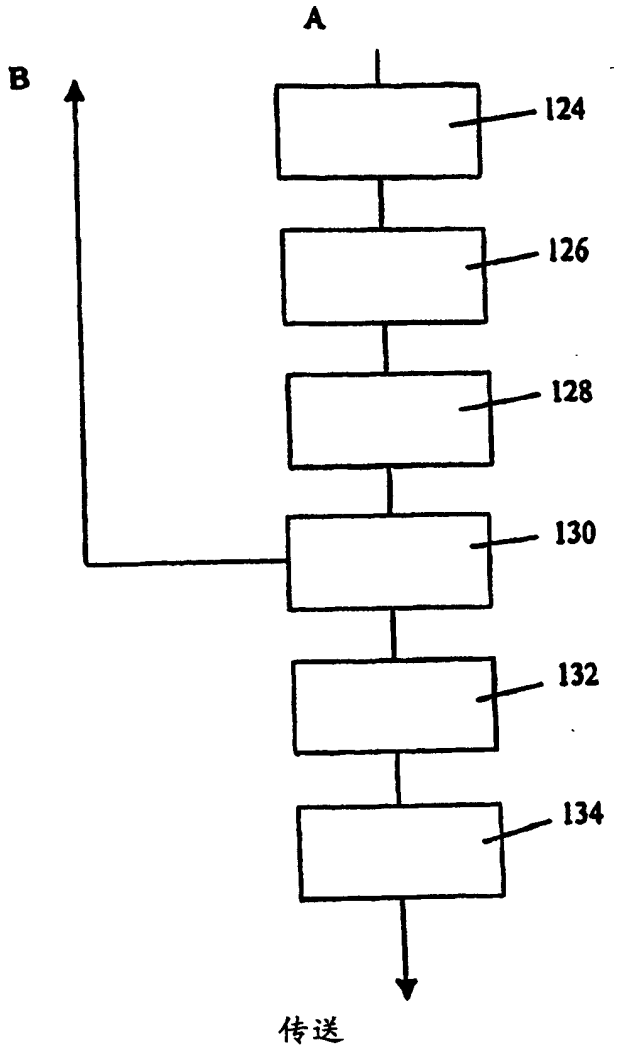


图 11 续