[19] 中华人民共和国国家知识产权局

[51] Int. Cl[7]
G06F 9/30
H04L 12/56 G06F 9/38
G06F 9/46

[12] 发明专利申请公开说明书

[21] 申请号 01809008.7

[43] 公开日 2003 年 8 月 20 日

[11] 公开号 CN 1437724A

[22] 申请日 2001.3.2 [21] 申请号 01809008.7
[30] 优先权
[32] 2000. 3. 3 [33] US [31] 60/186,782
[86] 国际申请 PCT/US01/06901 2001.3.2
[87] 国际公布 WO01/67237 英 2001.9.13
[85] 进入国家阶段日期 2002.11.4
[71] 申请人 坦诺网络公司
地址 美国麻萨诸塞州
[72] 发明人 T·胡西 D·W·蒙雷
A·N·索德

[74] 专利代理机构 中国专利代理(香港)有限公司
代理人 吴立明 王 勇

权利要求书 5 页 说明书 13 页 附图 8 页

[54] 发明名称 使用内部处理器存储空间的高速数据处理

[57] 摘要

通过把处理器的操作限制在其内部寄存器文件内以便减少由处理器执行的指令计数就能够在数据处理系统中实现显著的性能改进。 使用独立于处理器的直接存储器存取能够把小到足以装入该内部寄存器文件内的数据传送到该内部寄存器文件内，并且能够从中取出执行结果，从而使处理器能够避免执行装入及存储指令来操作外部存储的数据。 此外，数据以及处理活动的执行结果还可以由处理器在该内部寄存器文件内完全地存取和操作。 与多处理器的标准以及其指令集结合的，指令计数上的降低使得能够实现在可管理的复杂性及成本水平上的高度可调节，高性能的对称多处理系统。

ISSN1008-4274

1、 一种处理包的方法，该方法包括步骤：

接收包；

识别数据包的包头部分；

把包头传送到一处理器所能存取的寄存器文件；以及

处理包头而不需由处理器调用装入指令及存储指令中的至少一个。

2、 权利要求 1 的方法，其中不需调用装入指令及存储指令中的至少一个而执行传送步骤。

3、 权利要求 1 的方法，进一步包括步骤：

把包分成包头部分和包体部分；

使用直接寄存器存取把包头传送到该寄存器文件；以及

把包体传送到一输出缓冲器。

4、 权利要求 3 的方法，进一步包括步骤：

选择一个用于包的传输的输出端；

在该输出缓冲器内把处理过的包头与包体结合起来；以及

把结合过的包从该输出缓冲器转发到从其传输的选定输出端。

5、 权利要求 1 的方法，进一步包括步骤：

提供多个相同的执行一公共指令集的处理器，每一个处理器本地地给处理器存储了该指令集；

从多个处理器之中选择一个处理器来处理包头；以及

使该选定的处理器处理包头。

6、 权利要求 5 的方法，其中由对输入端上的包接收响应的一个状态机来执行选择处理器的步骤。

7、 权利要求 5 的方法，其中由至少一个被配置为把包头写入该选定处理器所能存取的寄存器文件内的至少一个固定位置的状态机来执行使该选定处理器处理包头的步骤。

8、 权利要求的方法，进一步包括步骤：把一公共指令集下载到多个处理器的每一个处理器内的一指令存储器上。

9、 一种处理在通信网络上接收的包的包头的方法，该方法包括步骤：

把包头传送到一寄存器文件内的至少一个固定位置；

提供一个与该寄存器文件有关的处理器，该处理器重复执行在一无限循环中的一条指令，该指令存储在与该处理器有关的指令存储器内的第一已知位置上；

使该处理器响应于包头的传送而执行从该指令存储器内的第二已知位置开始的指令；

根据从该指令存储器内的第二已知位置开始的指令处理在该寄存器文件内的至少一个固定位置中的包头；以及

在完成包头的处理时复位该处理器以重复执行存储在指令存储器内的第一已知位置上的指令。

10、权利要求 9 的方法，其中处理步骤包括处理包头而不需调用装入指令及存储指令的至少一个。

11、权利要求 9 的方法，进一步包括步骤：

在与该通信网络耦合的一输入端上接收包；

从与该输入端有关的多个候选处理器中选择处理器；

把包分成包头和包体；以及

通过执行与该寄存器文件耦合的一状态机所发出的 DRA 命令来把包头传送到与所选定处理器有关的寄存器文件内的至少一个固定位置。

12、权利要求 11 的方法，进一步包括步骤：把一公共指令集下载到多个候选处理器中的每一个处理器内的一指令存储器上。

13、一种用于处理在通信网络上接收的包的包处理系统，该系统包括：

被配置为在通信网络上接收包的一输入端；

与该输出端有关的一处理器；

该处理器所能存取的一寄存器文件；以及

与该输入端，处理器，以及寄存器文件耦合的一入口元件，该入口元件被配置为通过引用 DRA 命令而把包的至少一部分传送到寄存器文件，

其中该处理器响应于该 DRA 命令并且不需调用装入指令及存储指令中的至少一个来处理寄存器文件中的包的该至少一部分。

14、权利要求 13 的包处理系统，其中该入口元件被配置为从与该

输入端有关的多个候选处理器中选择处理器。

15、权利要求 14 的包处理系统，进一步包括多个指令存储器，该多个指令存储器中的每一个与多个候选处理器中的对应一个有关，其中该多个指令存储器含有一个相同的指令集。

16、权利要求 13 的包处理系统，其中包的该至少一部分对应于包头。

17、权利要求 16 的包处理系统，其中该入口元件包括被配置为把包头写入寄存器文件内的一固定位置的一个状态机。

18、一种用于处理在通信网络上接收的包的包头的包处理系统，该系统包括:

与通信网络耦合的一输入端;

与该输入端耦合并被配置为接收及分析包以获得包头的一入口元件;

与该入口元件耦合并被配置为把从该入口元件接收的包头存储在一至少一个固定位置上;

被配置为从至少一第一和第二地址返回指令的一指令存储器; 以及

与该入口元件，寄存器文件，以及指令存储器耦合的一处理器，该处理器重复执行存储在该指令存储器的第一个上的指令，其中该处理器执行从该指令存储器的第二地址上开始的指令以响应于来自该入口元件的信号处理寄存器文件内的包头。

19、一种信息处理系统，包括:

一处理器，具有一内部寄存器文件空间以及用于操作数据的一个单元;

一入口元件，用于把未处理的数据传递到内部寄存器文件空间; 以及

一出口元件，用于从内部寄存器文件空间取出处理过的数据，

其中处理器的操作被限制在操作内部寄存器文件空间内的数据。

20、权利要求 19 的系统，进一步包括至少一个管理入口及出口元件的操作的状态机，并且响应于内部寄存器文件空间内的指令，该状态机根据指令使用对该内部寄存器文件空间的直接存取使数据移入或移出内部寄存器文件空间。

21、权利要求 20 的系统，进一步包括从通信网络接收数据的一网络接口，该接口向入口元件提供接收到的数据。

22、一种信息处理方法，该方法包括步骤:

提供一个具有一内部寄存器文件空间以及用于操作数据的一个单元的处理器；以及

使用对该内部寄存器文件空间的直接存取来把未处理的数据传递到该内部寄存器文件空间并从内部寄存器文件空间取出处理过的数据，处理器的操作被限制在操作该内部寄存器文件空间内的数据。

23、权利要求 22 的方法，进一步包括步骤:

提供至少一个状态机使用对该内部寄存器文件空间的直接存取来管理向该内部寄存器文件空间的数据传递以及从内部寄存器文件空间的数据移出；以及

通过把一个值写入一控制寄存器来使该处理器发信号给状态机，该状态机响应于该值并根据状态机逻辑来执行该直接存取。

24、权利要求 22 的方法，其中该未处理的数据由具有线数据速率的通信网络发起，该处理器以和该线速率相等的速率处理数据。

25、权利要求 24 的方法，其中该未处理的数据为包格式。

26、一种处理含有一包的暂时序列的包流的方法，该方法包括步骤:

提供多个执行一公共指令集的相同的处理器，每一个处理器本地地给处理器存储了该指令集;

接收包;

对每个包，（i）识别数据包的包头部分，（ii）根据处理器的有效性从多个处理器之中选择一个处理器来处理包头，以及（iii）使该选定的处理器使用本地存储的指令处理包头；以及

根据该暂时序列组合处理过的包以重构包流。

27、权利要求 26 的方法，其中该多个处理器物理地放置在多个集成电路上。

28、一种用于处理含有一包的暂时序列的包流的系统，该系统包括:

多个执行一公共指令集的相同的处理器，每一个处理器包括含有该指令集的本地指令存储器;

用于接收包的一输入端;

与输入端以及处理器耦合的一入口逻辑单元，对于每个包，该入口逻辑单元被配置为，（i）识别数据包的包头部分以及（ii）根据处理器的有效性从多个处理器之中选择一个处理器来处理包头，该选定的处理器通过使用本地存储的指令处理包头以响应于入口逻辑单元;以及

一出口逻辑单元，用于根据该暂时序列组合处理过的包以重构包流。

29、权利要求 28 的系统，其中该多个处理器物理地放置在多个集成电路上。

使用内部处理器存储空间的高速数据处理

相关申请的前后参照

这里要求申请日为2000年3月3日，申请号为60／186，782的美国临时专利申请的优先权和权益，其全部内容在此并入作为参考。

发明领域

本发明总体来说涉及信息处理，并且具体地涉及发生在处理器的内部元件内的处理活动。

发明背景

数据处理典型地包括从存储器检索数据，处理该数据，以及把该处理活动的结果存回该存储器。支持该数据处理的硬件体系结构一般地控制信息以及信息处理系统的单个硬件单元之间的控制的流程。这一硬件单元的一种是处理器或处理引擎，其包含算术和逻辑处理电路，通用或专用寄存器，处理器控制或定序逻辑，以及使这些元件互相联系的数据通路。在某些实现中，可以把处理器配置为作为一定制设计的集成电路实现的或在一专用集成电路（ASIC）内实现的独立中央处理单元（CPU）。该处理器具有内部寄存器用于与由一组指令定义的操作一起使用。这些指令一般地存储在指令存储器内并指定一组在该处理器上有效的硬件功能。

当实现这些功能时，处理器一般从该处理器外的一个存储器检索“瞬态”数据，通过执行“装入”指令顺序地或随机地把数据部分装入其内部寄存器中，按照指令处理数据，并且之后使用“存储”指令把处理过的数据存回到外部存储器中。除了把瞬态数据装入内部寄存器以及把执行结果移出内部寄存器外，在瞬态数据的实际处理期间也频繁地使用装入及存储指令以便访问完成处理活动（例如，访问状态及命令寄存器）所需的附加信息。对外部存储器的频繁装入/存储访问一般是低效的，这是因为处理器的执行能力实质上比其外部接口能力快。因此，处理器常常闲置而等待把所存取的数据装入其内部寄存器文件中。

这一无效能够特别地在工作于通信系统内的设备上受到限制，因

为净效应将制约设备的整个数据处理能力以及，除非某些数据被除去而不是传送，网络自身的最大信息率。

发明概述

本发明考虑到对外部存储器的频繁访问对于处理足够小的被包含在被分配来处理数据组的本地寄存器文件空间内的数据组不是必需的。因此，本发明结合了至少部分地，独立于处理器而执行的数据存取技术并且其避免了处理器对装入及存储指令的执行。

在一实施例中，结合了本发明方面的信息处理系统和方法把分配来处理一数据组的处理器的操作限制在处理器的内部寄存器文件内。该信息处理系统包括一处理器，一入口元件，以及一出口元件。入口元件从一接口接收未处理的数据给对应于，例如从通信网络接收数据的网络接口的数据源。入口元件通过直接访问内部寄存器文件空间来把该未处理的数据，或其中的部分数据输送给内部寄存器文件空间。一个用于在处理器（例如，一算术逻辑单元）内操作数据的单元响应于向该处理器的寄存器文件的传送而操作及处理该数据并把其运行完全控制在其内部寄存器文件空间内。在处理活动完成时，入口元件直接访问并从内部寄存器文件空间中取出未处理的数据。或者，一中间状态机直接访问该未处理的数据并把它传送给出口元件。

在本发明的一个方面中，一个或多个状态机被包含在内并管理入口和出口元件的操作。一个或多个状态机还可以被包含在处理器内。状态机直接访问处理器的内部寄存器文件空间以便向该处输送数据或从中取出数据。在一实施例中，状态机的数据传送活动响应于 a）入口元件上的未处理数据的接收，b）用处理器逻辑表示未处理数据传送到该处理器的寄存器文件空间内的信号，以及/或 c）存储在一逻辑元件，如命令寄存器内的值的变化而被启动。

本发明的益处能够在许多信息处理系统，如集中于图像处理，信号处理，视频处理，以及网络包处理的这些系统中实现。作为一个实例，本发明能够体现在一通信设备，如路由器内以实现网络服务如路由处理，路径确定，以及路径切换功能。路由处理功能确定对于一个包所需的路由类型，而路径切换功能允许路由器在一接口上接受一个包并在第二接口上转发该包。路径确定功能选择用于转发该包的最适合的接口。

通信设备的路径切换功能能够在一个或多个结合了本发明的方面的转发引擎 ASIC 内实现，以支持在通信设备的多个接口之间的包传送。在此说明性的实施例中，由与该通信设备的网络接口的一特定输入端有关的入口逻辑经由通信网络来接收包数据。然后由入口逻辑从与接收端口有关的候选处理器组合（pool）中选择一处理器以处理该包。

一旦已经分配处理器，该包被分成为头部和包体部分。通过被配置成使用直接存储器/寄存器访问并且无需处理器引用装入或存储指令的入口逻辑的至少一个状态机来把包头写入一存储器元件内的适当位置，如与分配的处理器有关的内部寄存器文件。包体部分被写入一输出缓冲器。然后该处理器根据本地存储的指令来处理包头（再次，无需引用装入或存储指令）并把处理过的包头传送到所选的输出缓冲器，在输出缓冲器内与包体结合并且随后被从通信设备传送到用于传输的目的输出端。

在接收包头之前，所分配的处理器在一无限循环中重复执行存储在处理器的指令存储器中的第一已知位置/地址上的一条指令。处理器中的硬件检测地址 0 为用于电路指令返回的，而不是来自与处理器耦合的指令存储器的“特定”地址。当包头被从入口逻辑传送到处理器时，一控制信号向该处理器表明头部传送正在进行中。当此信号被激活时，处理器硬件迫使处理器程序计数器为一非特定地址（例如，地址 2），其终止无限循环的执行。在完成包头的传送时，处理器开始执行从其指令存储器的地址 2 开始的指令。一旦完成包处理活动，该处理器被复位（例如，把程序计数器置为地址 0）以重复执行在上述特定地址上的指令。

在此方式中，包头被直接写入处理器的寄存器文件，处理器无需要求任何交互或先前的知识直到准备处理包头时为止。与包的状态或特性有关的其它信息（例如，长度）也能够本地地存储在使用类似过程的寄存器文件内以使处理器不必访问外部资源以获得此信息。

为简化用于多个处理器的编程模块，能够为各个包分配一个单独的处理器，其中每个处理器被配置为执行在其各自的指令存储器内的一组公共指令。分配足够的处理器以保证能够以通信网络的电缆/线路速率（即，网络接口的最大比特率）来处理包。当把本发明的各方面

并到一 ASIC 内的多个处理器中时所实现的减少的指令集减小了 ASIC 的模具大小，从而允许在 ASIC 内的许多处理器中的较大密度而无需遭遇技术障碍以及导致这一 ASIC 制造的不利极限。本发明的 ASIC 实现还是可调节的，例如，通过增大处理器的时钟速率，通过给 ASIC 增加更多的处理器，以及通过从多个 ASIC 聚集处理器（具有公共指令集）组合。

在一实施例中，本发明能够用在对称多处理（SMP）系统中，展示一精简指令系统计算机（RISC）结构，以处理在通信网络上所接收的包。SMP 系统包括具有公共软件的作为一组合运行的多个相同的处理器，这些处理器中的任何一个适合处理一特定的包。把每个输入包分配给该组合内的一个有效处理器，并且这些处理器使用公共指令集来并行处理这些包。SMP 系统重构处理过的包流，因此其显示出正确的包顺序。

附图的简要说明

当结合附图，从下面的详细说明中将更为容易地理解以上论述，其中：

图 1 示意地说明了把一通信网络耦合到其它网络，如 LANs，MANs，以及 WANs 的通信设备；

图 2 示意地说明了根据本发明一实施例的安装在图 1 的通信设备内的网络接口卡的几个元件；

图 3 示意地说明了根据本发明一实施例的构成图 2 的网络接口卡的一部分的转发引擎的几个元件；

图 4 提供了根据本发明一实施例的当运行图 3 的转发引擎时所执行步骤的流程图；

图 5 示意地说明了根据本发明一实施例的执行直接存储器以及直接寄存器存取的图 3 的转发引擎的入口逻辑和处理器的几个元件；

图 6 提供了根据本发明一实施例的在图 5 的入口逻辑和处理器的运行期间所执行步骤的流程图；

图 7 示意地说明了根据本发明一实施例的构成图 5 的处理器的更详细的元件组；以及

图 8 提供了根据本发明一实施例的当运行图 7 中描述的处理器元件时所执行步骤的流程图。

发明的详细说明

典型的微处理器执行装入和存储指令以把表示存储在处理器外部的存储元件内的数据结构的数据的临时图像装入到处理器的本地寄存器文件中用于进一步的执行。作为这里所使用的，术语“本地寄存器文件”意指在操作数据中可用的处理器的内部结构内的寄存器全体。“寄存器” 指的是存储元件的相异组，如 D 触发器。根据处理器设计，寄存器文件空间能够由存储器与触发器的组合构成。在任何事件中，一般都利用提供了多个可独立存取的读以及写端口的高速存储器元件来实现寄存器文件。在软件程序的执行期间，典型的处理器执行相对大量的装入/存储指令以把数据从外部存储器移到本地寄存器文件以及把执行结果从本地寄存器文件移到外部存储器。对外部存储器的这些频繁存取是被强迫的，这是因为将被处理的数据组过长以致不能装到本地寄存器文件的执行空间内。

本发明认识到对外部存储器的频繁访问对于处理足够小的（例如，128 至 512，8 位数据元素）可全部放在本地寄存器文件空间内的数据组不是必需的。如以下所详细描述的，本发明结合了直接存储器存取（DMA）与直接寄存器存取（DRA）技术以把数据和执行结果放置在处理器的寄存器文件内外而无需处理器执行指令，如装入和存储指令，以移动数据。在这里，DMA 指的是使用一个或多个状态机来把数据块移入或移出独立于处理器的内部或外部存储器。类似地，DRA 指的是 DMA 的特殊类型，即，包含了一个或多个数据块移入或移出独立于处理器的处理器寄存器文件空间的一种 DMA。在一个实施例中，寄存器文件区域被分配为具有两个写端口和三个读端口（与具有一个写端口和两个读端口的标准三端寄存器文件空间相反）的五端寄存器文件空间以便方便直接寄存器文件存取。此方法避免了相对慢的（与寄存器文件内的操作相比）对外部存储器的存取，避免了存储器等待状态，并减少了处理器指令集的大小。因此并且除了较大地增加一单独的处理器的性能外，含有这些处理器的专用集成电路（ASIC）的模具大小以及能耗能够降低并且 ASIC 内的处理器的总数能够大大地增加而不招致无法承受的花费。

尽管出于处理在网络上接收的包的目的，在下文中将把本发明描述为在一通信设备的网络接口卡中实现，但是该特殊实现仅仅是一说

明性的实施例并且本技术领域的技术人员将认识到能够从所要求的本发明得益的任何数量的其它实施例及应用。为了举例而无限制，本发明能够有益于包含相对小的数据集，如存在于图像处理，信号处理，以及视频处理中的这些数据集的信息处理应用。本发明还能够实现于大范围的各种网络通信设备（例如，转换器和路由器）以及其它信息处理环境中。

参见图 1，通信设备 150 经由通信链路 112 从通信网络 110 接收信息（例如，以包/帧，信元，或 TDM 帧的形式）并把所收到的信息传送到不同的通信网络或分支如局域网（LAN）120，城域网（MAN）130，或广域网（WAN）140 或是传送到本地附属末端站（未示出）。通信设备 150 能够包含多个网络接口卡（NICs），如 NIC 160 和 NIC 180，每一个都具有一系列输入端（例如，162，164 及 166）和输出端（例如，168，170 和 172）。输入端 162，164 及 166 从通信网络 110 接收信息并把它们传送到处理这些包以及使它们为在输出端 168,170 和 172 之一传输而做好准备的多个包处理引擎（未示出），输出端 168，170 和 172 对应于含有该末端站的通信网络如 LAN 120，MAN 130，或 WAN 140。

参见图 2，体现了本发明方面的网络接口卡（NIC）160 包括输入端 162，164，166，包处理或转发引擎 220，地址查找引擎（ALE）210，统计模块 230，排队/解除排队模块 240，以及输出端 168，170，172。NIC 160 在输入端 162，164，166 从基于包的通信网络 110（图 1）接收数据。转发引擎 220，与 ALE 210 一起，通过查找与目标有关的适宜的输出端 168，170，172 来确定包的目标输出端，以及把转送矢量预先挂起到包上以协助把它们路由到适宜的输出端。

被改变的包被传送到排队/解除排队模块 240，其中转送矢量被用来把包组织到与特定目标输出端 168，170，172 有关的队列中。然后各个包的转送矢量被除去并且把包排定以用于向选定的输出端 168，170，172 的传输。接着把包从选定的输出端 168，170，172 传送到通信网络如 LAN 120，MAN 130，或 WAN 140。在一实施例中，NIC 160 的排队/解除排队模块 240 经由全网络（full-mesh）互连（未示出）接收该改变包因此它能够把在安装在通信设备 150 内的任何 NIC 160，180 的输入端上最初接收的包，包括由其自己的 NIC 160 的输入端 162，

164，166 接收的包集中到其自己的 NIC 160 的输出端 168，170，172 的一个或多个上。在另一实施例中，由转发引擎 220 把在输入端 162，164，166 上接收的包直接传送到排队/解除排队模块 240。

参见图 3 和图 4，转发引擎 220 的结构的一说明性实施例包括入口逻辑 310，ALE 接口 350，统计接口 360，出口逻辑 370，以及在 320，330，340 上代表性地示出的一个或多个处理器。运转中，对应于一个包的数据在通信网络 110 上发送并在与通信网络 110 耦合的 NIC 160 或 180 的一个特定输入端 162，164，或 166 上被接收（步骤 410）。然后从与输入端 162，164，或 166 有关的处理器（代表性地表示在 320，330，340 上）组合中选出处理器 330（步骤 420）。一旦已经分配了处理器 330，由入口逻辑 310 把包分成包头及包体部分（步骤 430）。使用直接寄存器存取把包头写入与处理器 330 有关的寄存器文件 710（图 7）内的一特定位置中，使用直接存储器存取把包体写入出口逻辑 370 中的一输出缓冲器中（步骤 440）。然后处理器 330 根据本地存储的指令处理包头（步骤 450）并把处理过的包头传送到出口逻辑 370，在出口逻辑 370 包头与包体重新结合（步骤 460）。

处理器 330 可以执行这样的任务如通过检查包头的完整性来处理包头，检验其校验和，经由统计接口 360 访问统计模块 230 以提供被用来向转发引擎 220 外部的模块报告涉及此包头的处理活动的统计，以及经由 ALE 接口 350 与 ALE 210 通信以获得用于与该包的目标有关的输出端 168，170，172 之一的路由信息。可以在此时做出附加的网络特定（例如，IP，ATM，Frame Relay，HDLC，TDM）包处理。在该处理活动终了时，处理器 330 修改包头以包括指定 NIC 160 的一特定输出端 168，170，172 的路由信息（例如，通过把转送矢量预先挂起到包头）。然后把改后的包头写入转发引擎 220 的出口逻辑 370，在出口逻辑 370 中该包头被接着路由到如上所述的排队/解除排队模块 240。

ALE 接口 350，统计接口 360 以及出口逻辑 370 是在转发引擎 220 内的可在处理器 320，330，340 之中被共享的资源。在转发引擎 220 设置了一个判优装置（未示出）以在访问这些资源 350，360，370 的处理器 320，330，340 之间判优。在一实施例中，当把处理器 330 分配给包时，用于处理器 330 的处理器标识符，诸如处理器号码被传给

上述标识的三个共享资源 350，360，370 中的每一个。然后这些共享资源 350，360，370 中的每一个把该处理器号码写入一 FIFO 内，其最好具有与转发引擎 220 内的处理器总数相等的深度。各个共享资源 350，360，370 中的逻辑访问其各自的 FIFO 以确定处理器 320，330 或 340 中的哪一个应当是下一个准许访问资源的处理器。一旦所准许的处理器完成其对特定资源 350，360，370 的访问，被访问的资源就读取其下一个 FIFO 入口以确定将向其发出准许的下一个处理器。

更具体地并且参见图 5 和 6，在转发引擎 220 内的包数据的接收，操作以及传送主要由多个 DMA 及 DRA 状态机来处理。在一说明性实施例中，这些状态机包含在入口逻辑 310 以及处理器 330 内。在此说明性实施例的操作期间，从 NIC 160 的输入端 162，164，166 之一接收包并把它存储在入口逻辑 310 内的接收-数据 FIFO（先进/先出缓冲器）之中（步骤 610）。接收-状态 FIFO 512 记录该包所到达的特定输入端 162，164，或 166 并为每个由转发引擎 220 接收的包保留一输入端号码的有序清单，该清单根据包被接收到的时间分类。

发出-DMA-命令状态机 514 检测何时接收-状态 FIFO 512 含有数据以及获取与从接收-状态 FIFO 512 接收包的输入端 162，164，或 166 有关的输入端号码（步骤 620）。之后发出-DMA-命令状态机 514 向分配-处理器状态机 516 发送含有包的端口号的处理器分配请求，装置 516 访问与那一端口号有关的分配-组合寄存器 518 以确定候选用于处理该包的一组处理器 320，330，340（步骤 630）。然后分配-处理器状态机 516 访问处理器-空闲寄存器 520 以确定由分配-组合寄存器 518 所标识的候选处理器 320，330，340 的任何一个是否可使用。分配-处理器状态机 516 接着从一组候选处理器 320，330，340 中分配一个可用处理器 330 以处理该包（步骤 640）并把分配准许以及处理器 330 的处理器号码发送给发出-DMA-命令状态机 514。

在收到与所分配的处理器 330 有关的处理器号码时，发出-DMA-命令状态机 514 向 DMA-执行状态机 522 发送含有该处理器号码的一执行信号/命令，装置 522 访问头部-DMA-长度寄存器 524 以获得将被发送给处理器 330 的接收的包的数量（即，包头的长度）（步骤 650）。然后 DMA-执行状态机 522 发出一 DMA 命令，其从接收-数据 FIFO 510 检索包头部分（对应于包头）并在 DRA 总线 526 上传送它，在该总线

上由包含在处理器 330 内的处理器-DRA 状态机 530 接收包头部分（步骤 660）。DMA-执行状态机 522 还发出从接收-数据 FIFO 510 检索包体并在另一条 DMA 总线 528 上传送它以由出口逻辑 370 的缓冲器（未示出）来接收的一条命令。处理器-DRA 状态机 530 接着把经由 DRA 总线 526 接收的包头数据直接写入从处理器 330 的寄存器文件空间 710（图 7）内的一固定地址位置（例如，地址 0）开始的寄存器文件区域（步骤 670）。然后处理器 330 处理包头（步骤 680）并把处理过的包头经由传送-DMA 状态机 532 传送给出口逻辑 370 用于与包体重新组合（步骤 690）。

更具体地并参见图 7 和 8，在处理器 330 内的包头的处理是更可取的以使处理器的指令及活动被限于数据操作以及构成于处理器本地寄存器文件 710 内的执行空间中的执行结果。在一说明性实施例中的处理器 330 的结构包括 Stats-接口状态机 704，ALE-接口状态机 706，处理器-DRA 状态机 530，传送-DMA 状态机 532，寄存器文件 710，算术逻辑单元（ALU）720，处理器控制模块 730，以及指令存储器 740。计算单元 725 由处理器控制 730 及 ALU 720 组成。

在此说明性实施例的运行期间以及当处理器 330 等待接收包头时，计算单元 725 连续地执行在指令存储器 740 内一特定地址（例如，地址 0）上的指令（即，在一无限循环中）（步骤 810）。处理器 330 内的硬件检测地址 0 为其中指令被从蚀刻在硅上的“电路”指令值返回而不是从存储在指令存储器 740 内的指令返回的一特定地址。在一可能的实现中，对特定地址 0 上的指令的访问返回“JMP 0”（或跳到地址 0 指令），从而使处理器 330 在那一地址上执行一无限循环。

当包头被从入口逻辑 310 传送到处理器的寄存器文件 710 时，来自处理器-DRA 状态机 530 的一控制信号向处理器控制模块 730 表示包头传送正在处理中（步骤 820）。在此信号被激活时，处理器控制模块 730 迫使处理器程序计数器（未示出）指定指令存储器 740 的一非特定地址（例如，地址 2）并因此使计算单元 725 从在特定地址 0 被执行的无限循环中跳出并等待直到该信号变成无效的（步骤 830）。计算单元 725 响应于该信号变为失效而开始在地址 2 上的指令的执行（步骤 840）。指令存储器 740 的地址 2 能够被构成为保持将被用于处理寄存器文件 710 内的包头的第一指令（即，地址 2 上的指令对应

于先前已被下载来处理包头的“真实”软件图像的开始）。当处理器-DRA 状态机 530 完成从寄存器文件 710 内的一固定位置开始的包头的写入时（当控制信号成为无效时发生），计算单元 725 继续正常地执行指令存储器 740 内的剩余指令（即，地址 2 以外）。指令存储器 740 内的特定指令指定了寄存器文件 710 内的位置。当完成对特殊包头的处理活动时，执行软件“跳到”地址 0，从而在无限循环中执行地址 0 上的指令。此项技术说明了如何触发处理器 330 以处理存储在寄存器文件 710 内的包头而不使用装入及存储指令的一个特殊实现。

在另一实施例中，所分配的处理器 330 保持空闲（即，不访问指令存储器或执行指令）直到它从外部状态机收到表示寄存器文件 710 已被完整的包头填充的信号时。然后计算单元 725 执行来自指令存储器 740 的代码以处理该包头。触发事件能够，例如，包括当控制信号变为无效。另一方面，当 DRA 传送已被启动，完成时，或当其正在进行时触发所分配的处理器 330。许多其它触发事件对于本领域的技术人员来说将是显而易见的。

如早先所论述的，处理器 330 在包头的处理期间访问处理器 330 外部的一个或多个共享资源（例如，见图 3，ALE 接口 350，统计接口 360，以及出口逻辑 370）。例如，处理器 330 通过 ALE 接口 350（图 3）与 ALE 210（图 2）相互配合来发出 ALE 210 的搜索并从其接收搜索结果。由处理器 330 执行的这些与 ALE 210 的交互不需处理器 330 执行装入及存储指令也会发生。

在一个方面以及当执行指令存储器 740 内的指令时，处理器 330 组成一个从寄存器文件 710 中的一预定地址开始的搜索关键字。计算单元 725 执行涉及把一个值写入 ALE-命令寄存器来规定发送给 ALE 210 的搜索关键字数据的数量的一条指令。该值有效地用作为用于处理器 330 的 ALE-接口状态机 706 的控制线并由此触发 ALE-接口状态机 706 以从 ALE-命令寄存器读取该值或其它数据，确定要被传送的数据量，以及利用独立于计算单元 725 的直接存储器存取把指定数据传送给 ALE 接口 350。当处理器 330 等待被返回的搜索结果时，它能够执行其它功能，如检验包头的网络协议（例如，IP）校验和。当来自 ALE 210 的搜索结果有效时，经由 ALE 接口 350 把它们发送给 ALE-接口状态机 706。ALE-接口状态机 706 使用一个或多个直接寄存器存取把搜索结果

写入寄存器文件 710 的预定位置并当写完成时发信号给计算装置 725。接着计算装置 725 响应于该搜索结果修改包头。

处理器 330 还能够通过把一地址以及长度值写入处理器 330 的统计-更新-命令寄存器（未示出）来发出统计更新命令。触发处理器 330 的统计-接口状态机 704 以从统计-更新-命令寄存器读取数据，确定源以及要传送的数据量，以及利用独立于计算单元 725 的直接存储器存取把指定数据传送到统计接口 360。

类似地，当处理器 330 已完成包头处理时，计算单元 725 把处理过的包头写入处理器 330 的发送-DMA 状态机 532，装置 532 利用独立于处理器 330 的直接存储器存取把该处理过的包头传送到出口逻辑 370 内的缓冲器（步骤 850）。当所有处理完成时，在处理器 330 内执行的软件跳回指令存储器 740 的地址 0 并开始执行先前论述的无限循环指令同时等待下一个包头到达（步骤 860）。

更具体地，在处理活动完成时，包头可以不必驻留在寄存器文件 710 的相邻区域以及因此计算装置 725 必须指定在寄存器文件 710 内的各个处理过的包头的位置。因此，计算装置 725 向移动-DMA-命令寄存器（未示出）发出指定各个处理过的包头的起始地址及长度的一个或多个写。这些写存储在一 FIFO 中，主要作为重编命令的清单。在获得用于所有不完全包头的数据后，计算装置 725 对发送-DMA-命令寄存器（未示出）写并指定与其它数据一起的包体长度。

写入发送-DMA-命令寄存器的值触发处理器 330 内的发送-DMA 状态机 532 开始根据存储在上述参照 FIFO 内的重编命令来开时包头的组合。然后发送-DMA 状态机 532 利用独立于计算单元 725 的直接存储器存取把所组合的包头与一些控制信息（包括包体长度）一起发送到出口逻辑 370。出口逻辑 370 把从发送-DMA 状态机 532 收到的处理过的包头与存储在出口逻辑 370 的 FIFO 中的包体连接在一起，接着把重构的包发送到如先前所述的排队/解除排队模块 240。

为了正确地重构包头与包体，处理器 330 从嵌入在包头本身内的数据获得整个包的长度并通过接收-数据 FIFO 510（图 5）从传送给处理器 330 的数据获得包头的长度（对应于写入图 5 的头部-长度寄存器 524 内的相同值）。基于这一信息，处理器 330 计算先前传送到出口逻辑 370 内的输出 FIFO 的包体数据量并把包体的长度指定为将通过发

送-DMA 状态机 532 发送给出口逻辑 370 的控制信息。在此方式中，处理器 330 能够把包体数据量指定为从出口逻辑 370 的输出 FIFO 抽出、将被加到由处理器 330 构成的新组合的包头上以重构修改数据包。为了正确地重构修改的数据包，准许处理器 330 按照与处理器 330 被分配的顺序相同的顺序来访问出口逻辑 370（并因此按照与包体被写入出口逻辑 370 的输出 FIFO 的顺序相同的顺序）。

本发明的各方面能够给输入包处理要求提供在计算资源分配上的极大的灵活性。假设为了说明性的目的在转发引擎 220 内有总共 40 个处理器 320，330，340，能够灵活地分配处理器 320，330，340 以符合众多输入/输出端结构的包处理需求。例如，在其中仅有一个单一逻辑输入端（即，端口 0）的 NIC 160 中，全部 40 个处理器 320，330，340 能够被分配为为该单一端口处理包。在此情况中，装入每个处理器 320，330，340 的指令存储器 740 内的代码图像应是相同的，从而使每个处理器 320，330，340 能够执行对于输入端的那一类型的相同算法。在另一种涉及四个逻辑输入端，每一个具有不同类型的网络接口的情况中，各种网络接口所需的处理算法可以不同。在此情形中，能够如下分配 40 个处理器：处理器[0-9]用于端口 0，处理器[10-19]用于端口 1，处理器[20-29]用于端口 2 以及处理器[30-39]用于端口 3。另外，能够下载 4 个不同的代码图像，其中每个单独的图像对应于一个特定的输入端。在又一种情况中，NIC 160 可以包括两个逻辑输入端，每一个具有不同的处理性能要求。在这一情况中，输入端之一可耗费 75%的入口总线带宽并具有要求 75%的处理器资源的包到达速率，而第二端占去剩余部分。为了支持这些性能要求，能够把 30 个处理器分配给输入端 0 以及把 10 个处理器分配给输入端 1。

用于包括了作为其转发引擎 220 的部件的多个处理器的 NIC 160，180 的编程模块，能够通过把一个单独的处理器分配给所收到的各个包来得到简化。此外，并且如上所述，通过含有本发明的系统而实现的减小了的模具大小允许在 NIC 160，180 的转发引擎 ASICs 内的附加处理器的包含关系，从而这就保证了能够按照网络 110 的线速传送包。通过在给定的转发引擎 ASIC 上增加更多的处理器，增加处理器的时钟速率，以及通过集成多个 ASIC 的处理组合，本发明是容易地可调节的。注意，在提供这一能力方面，本发明的硬件结构保持经由网络接口到

达的包的包顺序以使重组过的包能够以适当顺序发送到转发引擎外。

在通信设备 150 的 NIC 160 经由通信网络 110 以线速率接收包数据流，否则可能会压倒 NIC 160 的处理能力并导致包的减少以及降低服务质量时处理器组合集成技术会特别有益。该集成技术允许来自一个以上的转发引擎的空闲处理器的分配。例如，NIC 160 可包含多个转发引擎 ASIC，每一个具有能够被分配为处理在 NIC 160 上的任何输入端到达的包的处理器组。另一方面，存在于通信设备 150 内的其它 NICs 180 上的，除了转发引擎 ASIC 外的处理器组，能够被分配给经历繁重网络负载的 NIC 160。

尽管已参照特定细节对本发明进行了描述，但是并不意味着这些细节将被视为在本发明范围上的限制，除了作为以及它们被包含在附带权利要求书中的范围。

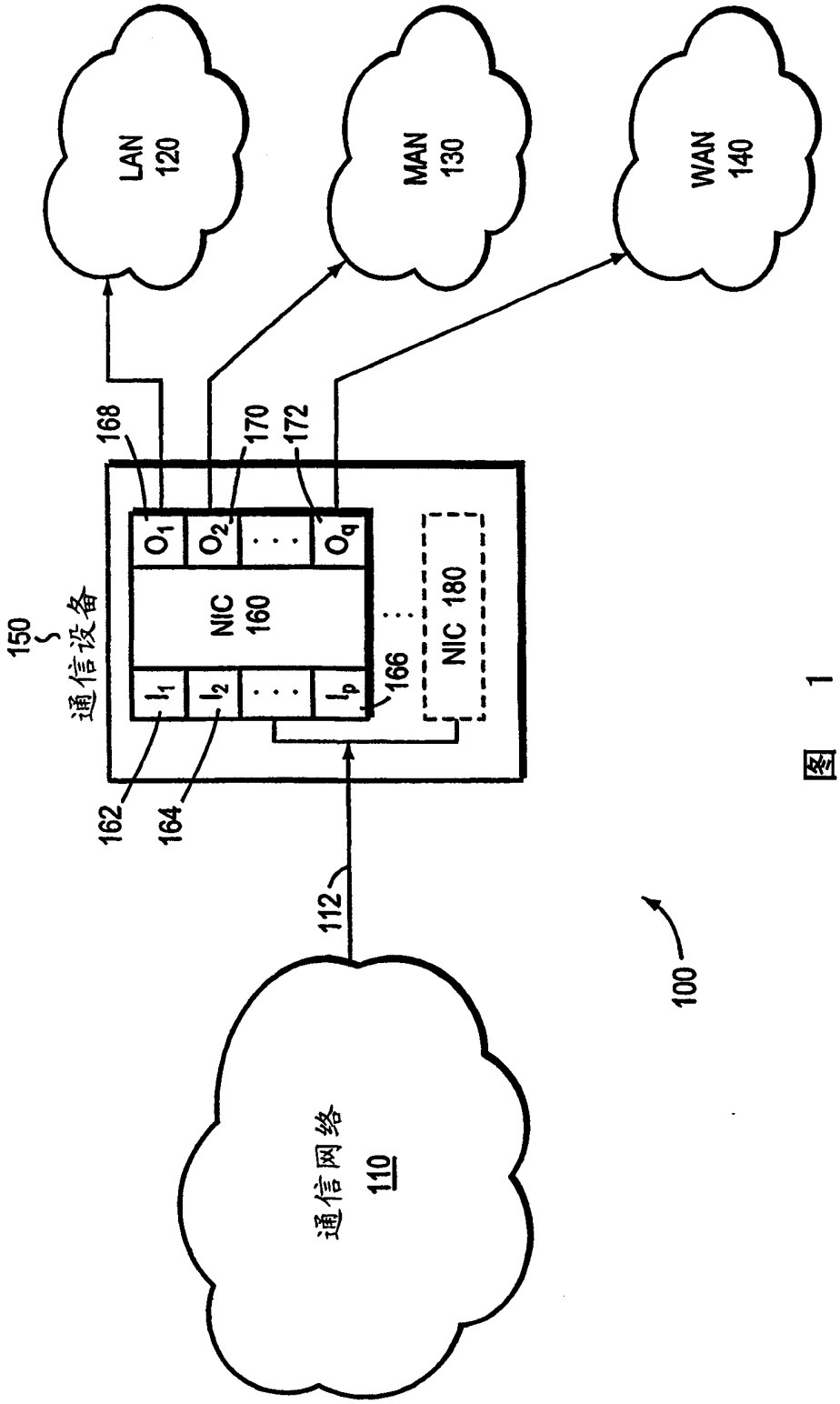
LAN
120
MAN
130
WAN
140
168
170
172
150
通信设备
O_1
O_2
O_q
NIC
160
NIC 180
166
I_1
I_2
I_p
162
164
112
通信网络
110
100

图 1

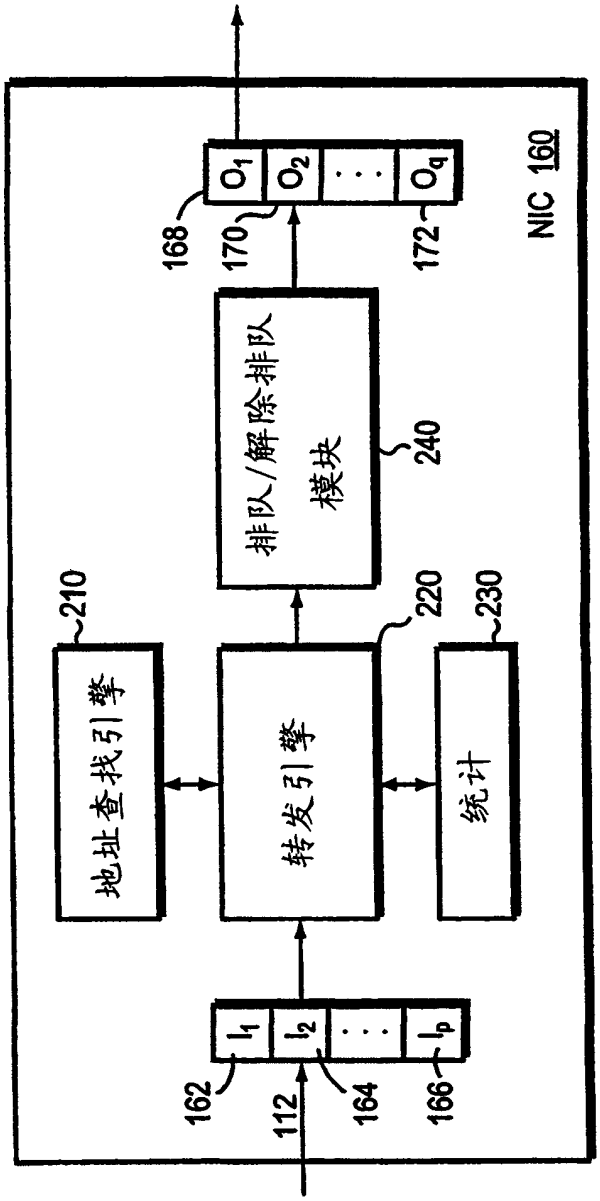
NIC 160
O_1
O_2
O_q
168
170
172
排队/解除排队模块
240
210
220
230
地址查找引擎
转发引擎
统计
I_1
I_2
I_p
162
112
164
166

图 2

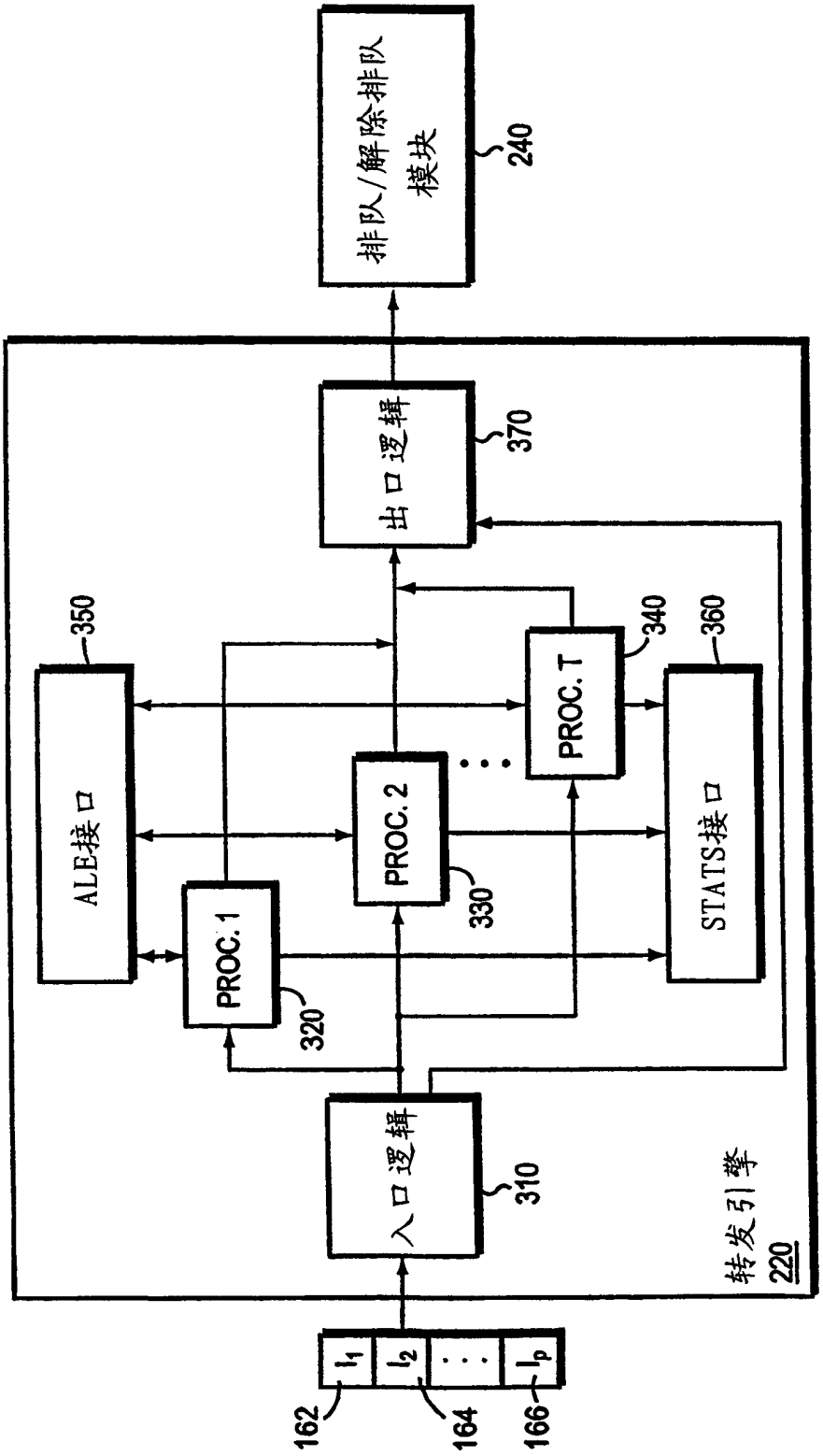
排队/解除排队模块
240
出口逻辑
370
ALE接口
350
PROC. T
340
360
PROC. 2
330
STATS接口
PROC. 1
320
入口逻辑
310
转发引擎
220
I_1
I_2
I_P
162
164
166

图 3

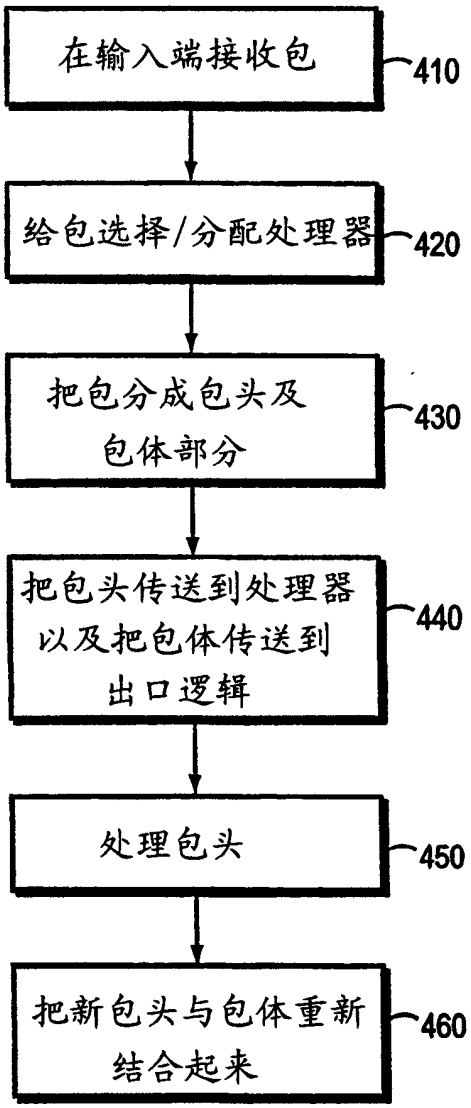
在输入端接收包
410
给包选择/分配处理器
420
把包分成包头及包体部分
430
把包头传送到处理器以及把包体传送到出口逻辑
440
处理包头
450
把新包头与包体重新结合起来
460

图 4

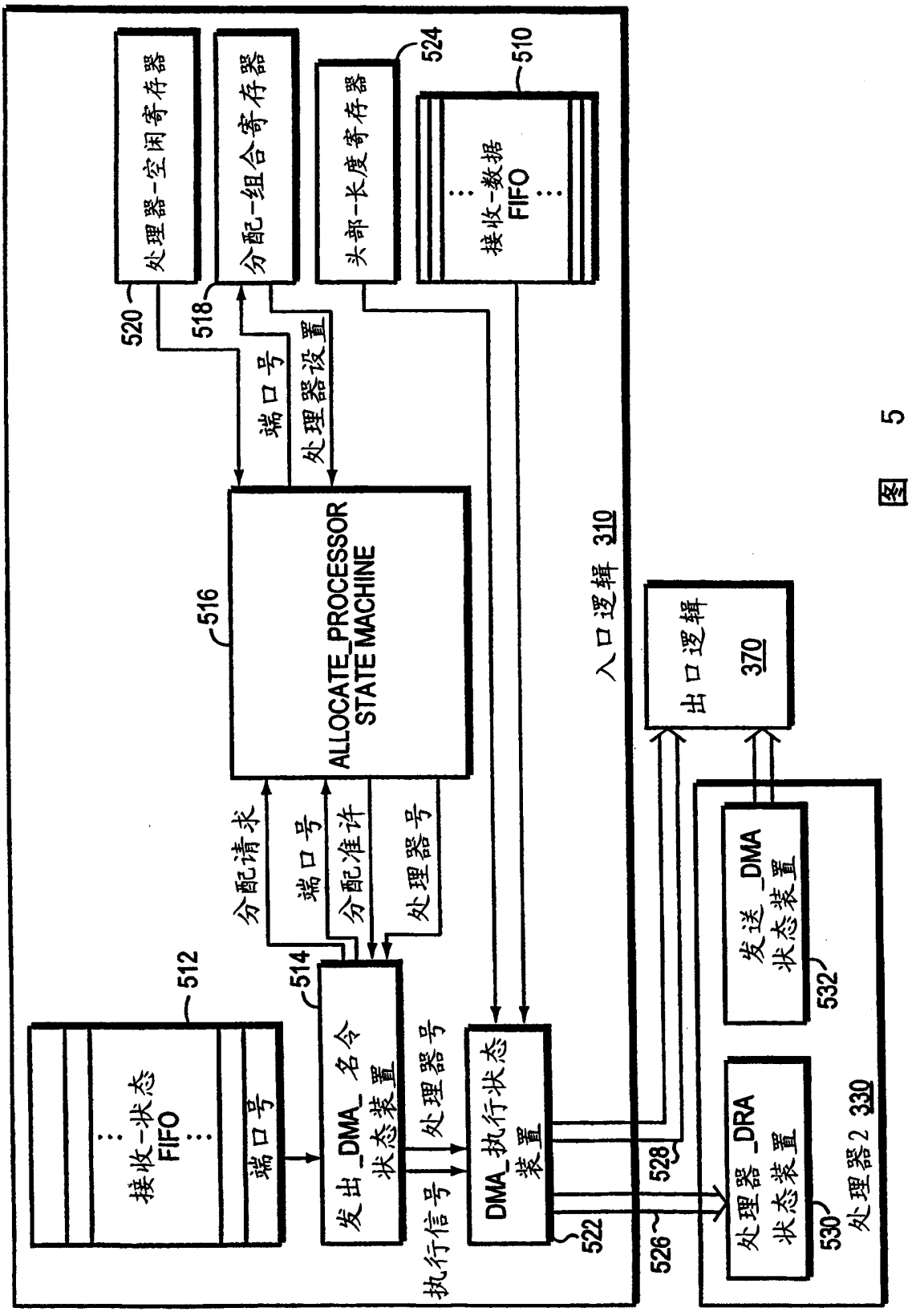
处理器-空闲寄存器
分配-组合寄存器
头部-长度寄存器
524
510
接收-数据 FIFO
520
518
端口号
处理器设置
516
ALLOCATE_PROCESSOR STATE MACHINE
入口逻辑 310
出口逻辑 370
分配请求
端口号
分配准许
处理器号
512
514
接收-状态 FIFO
端口号
发出_DMA_名令 状态装置
处理器号
DMA_执行状态 装置
执行信号
522
526
528
发送_DMA 状态装置
532
处理器_DRA 状态装置
530
处理器2 330

图 5

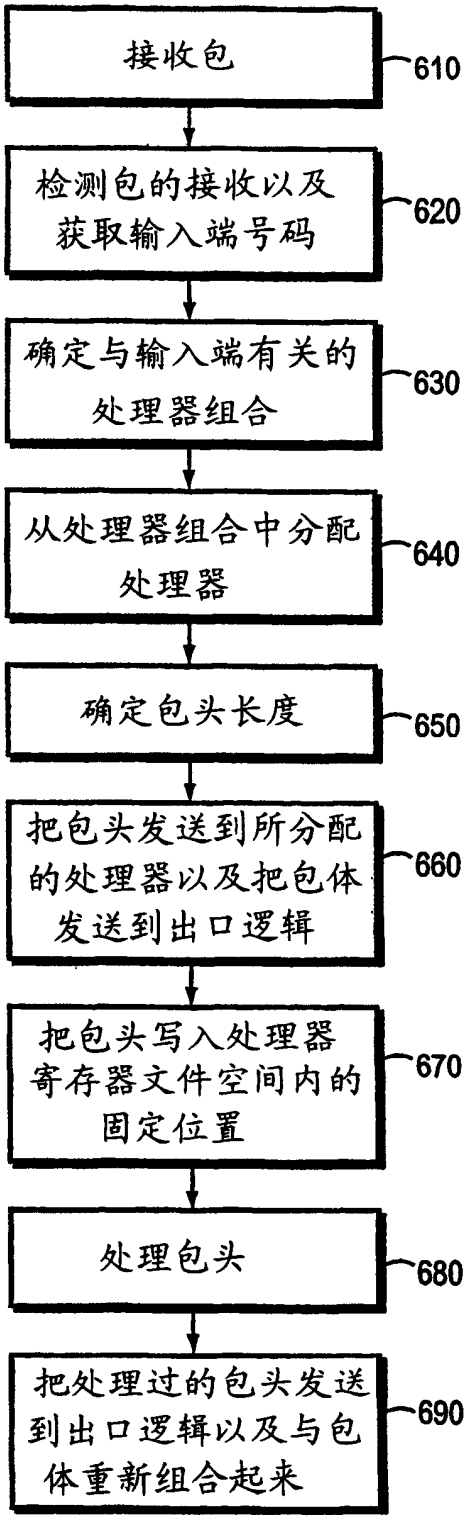
接收包
610
检测包的接收以及获取输入端号码
620
确定与输入端有关的处理器组合
630
从处理器组合中分配处理器
640
确定包头长度
650
把包头发送到所分配的处理器以及把包体发送到出口逻辑
660
把包头写入处理器寄存器文件空间内的固定位置
670
处理包头
680
把处理过的包头发送到出口逻辑以及与包体重新组合起来
690

图 6

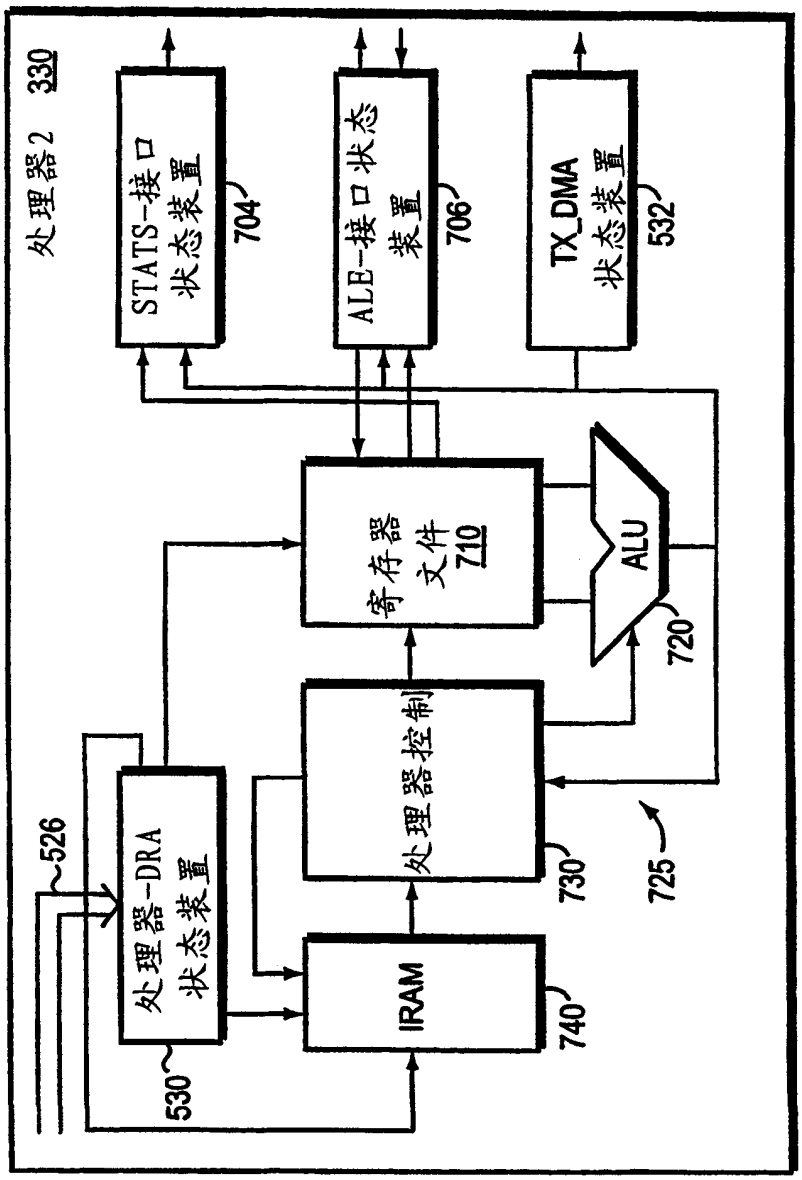
处理器2 330
STATS-接口状态装置 704
ALE-接口状态装置 706
TX_DMA状态装置 532
寄存器文件 710
ALU 720
处理器控制 730
725
526
处理器-DRA状态装置 530
IRAM 740

图 7

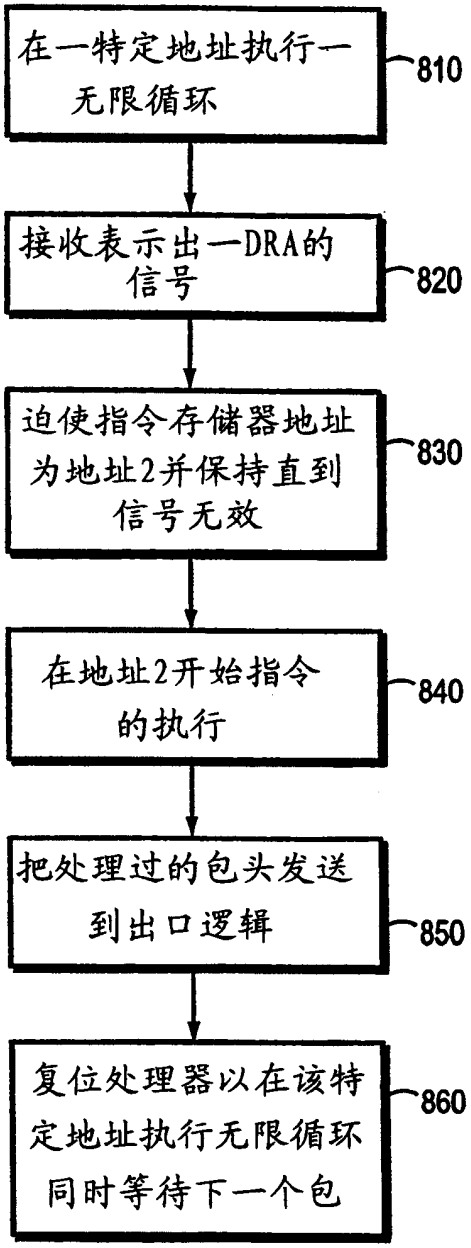
在一特定地址执行一无限循环
810
接收表示出一DRA的信号
820
迫使指令存储器地址为地址2并保持直到信号无效
830
在地址2开始指令的执行
840
把处理过的包头发送到出口逻辑
850
复位处理器以在该特定地址执行无限循环同时等待下一个包
860

图 8