

[19]中华人民共和国国家知识产权局

[51]Int. Cl⁷

G06F 9/30

G06F 12/08

[12] 发明专利申请公开说明书

[21] 申请号 00137005.7

[43]公开日 2001 年 7 月 11 日

[11]公开号 CN 1303044A

[22]申请日 2000.12.27 [21]申请号 00137005.7

[30]优先权

[32]2000.1.6 [33]US [31]09/478,308

[71]申请人 国际商业机器公司

地址 美国纽约

[72]发明人 詹姆斯·A·卡尔

[74]专利代理机构 中国国际贸易促进委员会专利商标事务所

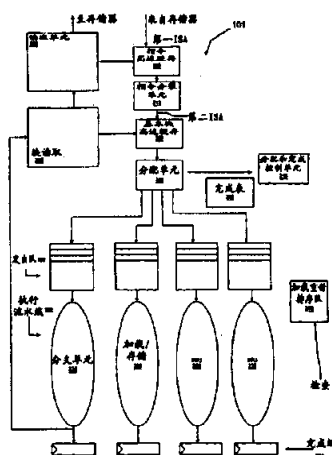
代理人 张 维

权利要求书 3 页 说明书 15 页 附图页数 8 页

[54]发明名称 使用基本块高速缓存的微处理器

[57]摘要

本发明披露了一种微处理器和一种相关的方法以及数据处理系统。所述微处理器包括指令分裂单元,其被构成用于接收第一个微处理器指令集。所述分裂单元把所述指令集组织成为指令组。在所述指令组中的每个指令共用公共的指令组标记。所述处理器还包括基本块高速缓存装置,其利用指令组格式被组织,并且被构成用于高速缓存由分裂单元产生的指令组。处理器的执行单元适用于执行指令组中的指令。



ISSN 1008-4274

权 利 要 求 书

1 一种用于执行微处理器指令的方法，包括以下步骤：

把接收的第一指令集转换为指令组；

在被设置的基本块高速缓存的入口中存储所述指令组，其中每个高速缓存装置的入口含有指令组；

发出在所述指令组中的指令进行执行；以及

响应在指令组中的指令执行期间产生的异常，只刷新悬挂在基本块高速缓存和最后一级之间的那些指令。

2 如权利要求 1 所述的方法，其中产生的异常包括分支误预测异常。

3 如权利要求 1 所述的方法，其中被接收的指令按照第一指令格式被格式化，并且在指令组中的指令按照第二指令格式被格式化。

4 如权利要求 3 所述的方法，其中第二指令格式比第一指令格式宽。

5 如权利要求 4 所述的方法，还包括对高速缓存装置中的每个入口分配一个指针，其中所述指针用于预测要被执行的下一个指令组。

6 如权利要求 5 所述的方法，还包括响应在一个指令组的执行期间检测到一个误预测分支更新相应于误预测的分支的高速缓存入口的指针。

7 一种微处理器，包括：

指令分裂单元，其构成用于接收第一微处理器指令集，并且把所述指令集组织成指令组；以及

基本块高速缓存装置，其构成用于高速缓存由分裂单元产生的指令组；以及

执行单元，其适用于执行指令组中的指令；

其中在指令组中的指令执行期间产生的引起刷新的异常，只刷新从基本块高速缓存分配的那些指令。

8 如权利要求 7 所述的处理器，还包括分配单元，其构成用于从基本块高速缓存中的指令组中检索指令，并把所述指令输送到发出队中。

9 如权利要求 7 所述的处理器，其中被接收的指令按照第一指令格式被格式化，并且第二指令集按照第二指令格式被格式化，其中第二指令格式比第一指令格式宽。

10 如权利要求 7 所述的处理器，其中在基本块高速缓存被构成用于在相应的基本块高速缓存的每个入口存储每个指令组。

11 如权利要求 10 所述的处理器，其中基本块高速缓存包括表示相应的基本块高速缓存入口的入口字段。

12 如权利要求 11 所述的处理器，其中基本块高速缓存的每个入口包括用于预测下一要被执行的指令组的指针。

13 如权利要求 12 所述的处理器，其中处理器被构成使得响应误预测的分支更新每个入口的指针。

14 一种包括至少一个处理器，存储器，输入装置和显示装置的数据处理系统，其中所述处理器包括：

指令分裂单元，其构成用于接收第一微处理器指令集，并且把所述指令集组织成指令组；

基本块高速缓存装置，其构成用于高速缓存由分裂单元产生的指令组；以及

执行单元，其适用于执行指令组中的指令；

其中在指令组中的指令执行期间产生的引起刷新的异常，只刷新从基本块高速缓存分配的那些指令。

15 如权利要求 14 所述的数据处理系统，还包括分配单元，其构成用于从基本块高速缓存中的指令组中检索指令，并把所述指令输送到发出队中。

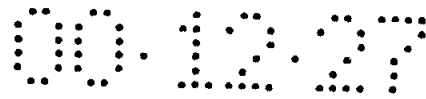
16 如权利要求 14 所述的数据处理系统，其中被接收的指令按照第一指令格式被格式化，并且第二指令集按照第二指令格式被格式化，其中第二指令格式比第一指令格式宽。

17 如权利要求 14 所述的数据处理系统，其中在基本块高速缓存被构成用于在相应的基本块高速缓存的每个入口存储每个指令组。

18 如权利要求 17 所述的数据处理系统，其中基本块高速缓存包括表示相应的基本块高速缓存入口的入口字段。

19 如权利要求 18 所述的数据处理系统，其中基本块高速缓存的每个入口包括用于预测下一要被执行的指令组的指针。

20 如权利要求 14 所述的数据处理系统，其中处理器被构成使得响应误预测的分支更新每个入口的指针。



说明书

使用基本块高速缓存的微处理器

本申请的主题和 1999, 10, 28 申请的序列号为 09/428399, 名称为“微处理器中的指令组组织和例外处理”的美国专利的主题相关, 其和本申请为同一受让人, 在此列为参考。

本发明一般涉及微处理器结构领域, 尤其涉及利用指令组结构、相应的高速缓存功能及其有用的扩展的微处理器。

微处理器技术已经达到千兆赫的性能, 对微处理器设计者的主要的挑战是利用达到最新技术发展水平的技术, 同时又保持和为利用特定的指令系统结构 (ISA) 操作而设计的大量的已安装软件兼容。为了解决这个问题, 设计者已经实现了“分层结构”的微处理器, 其适应于接收按照现有的 ISA 构成的指令, 并把接收的指令的指令格式转换为更适合于在千兆赫执行流水线中操作的内部 ISA。参看图 4, 图中示出了分层结构的微处理器 401 的选择的部分。在这种设计中, 微处理器 401 的高速缓存 410 接收并存储借助于取单元 402 从主存储器取来的指令。在指令高速缓存单元 410 中存储的指令按照第一 ISA (即其中写有由处理器 401 执行的程序的 ISA) 被格式化。然后从指令高速缓存 410 中检索指令并通过 ISA 转换单元 412 被转换为第二 ISA。因为从第一 ISA 到第二 ISA 的转换需要多个周期, 所以转换处理一般是流水线处理, 因而, 在任何给定的时刻, 可能有多个被从第一 ISA 转换为第二 ISA 的指令。然后输出被转换的指令, 以便在处理器 401 的执行流水线 422 中执行。取单元 402 包括分支预测逻辑 406, 其借助于预测分支决定的结果试图确定伴随着分支指令的将被执行的指令的地址。然后, 指令被预测性地发出并被根据分支预测执行。然而, 当一个分支预测错误时, 被悬在微处理器 401 的指令高速缓存 410 和结束级 432 之间的指令必须被刷

新。当误预测的分支引起系统刷新时，则招致作为流水线长度的函数的性能损失。必须被刷新的流水线的数量越大，分支误预测性能损失也越大。因为分层体系结构加于处理器流水线，并增加了在一个给定的时刻可能处于“飞行”中的指令的数量，所以和分层体系结构相关的分支预测损失可能成为处理器性能的一个限制因素。因此非常需要实现一种能够解决分支误预测性能损失的分层体系结构的微处理器。此外，还需要能够至少部分地解决由重复执行一段代码而引起的例外条件反复发生的解决方案。还需要一种能够操作实际上较大的发出队，而不牺牲检索下一个要执行的指令的能力。

上述的问题通过一种利用指令组和与所述指令组的格式匹配的高速缓存装置微处理器被大部分解决了。本发明的一个实施例提出了一种微处理器和一种相关的方法以及数据处理系统。所述微处理器包括指令分裂单元，其被构成用于接收第一个微处理器指令集。所述分裂单元把所述指令集组织指令组。在所述指令组中的每个指令共用公共的指令组标记。所述处理器还包括基本块高速缓存装置，其利用指令组格式被组织，并且被构成用于高速缓存由分裂单元产生的指令组。处理器的执行单元适用于执行指令组中的指令。在一个实施例中，当在执行指令组中的指令期间产生引起刷新的异常时，则所述的刷新只刷新那些已被从基本块高速缓存中分派的指令。通过只刷新那些已经到达基本块高速缓存的指令，处理器用不着刷新悬挂在分裂单元流水线中的指令。因为被刷新的指令较少，所以可以减少异常性能损失。在一个实施例中，接收的指令被按照第一指令格式格式化，并且按照第二指令格式对第二指令集格式化，其中所述第二指令格式比第一指令格式宽。基本块高速缓存被合适地构成，用于在基本块高速缓存的相应的入口中存储每个指令组。在一个实施例中，基本块高速缓存中的每个入口包括表示相应的基本块高速缓存入口的入口字段和预测下一个要被执行的指令组的指针。处理器最好被构成用于响应误预测的分支更新高速缓存的指针。

本发明还提出了一种利用指令历史信息和本基本块高速缓存一起来改善性能的微处理器、数据处理系统和方法。这种处理器适用于接收一个指令集，并把所述指令集组织指令组，然后，指令组被分派执行。在执行指令组时，记录表示和指令组相关的异常事件的指令历史信息。此后，响应指令历史信息修改指令的执行，以便阻止在随后的指令组执行期间发生异常事件。所述处理器包括存储装置，例如指令高速缓存，L2 高速缓存和系统存储器，分裂单元，和基本块高速缓存。分裂单元被构成用于从存储装置接收指令集。分裂单元适用于把指令集组织成指令组。分裂单元可以从第一指令格式到第二指令格式修改指令集的格式。基本块高速缓存的结构适用于存储指令组。基本块高速缓存包括相应于每个基本块高速缓存入口的指令历史字段。指令历史信息表示和指令组相关的异常事件。在优选实施例中，基本块高速缓存的每个入口相应于由分裂单元产生的一个指令组。处理器还可以包括完成表控制逻辑，其被构成用于当指令组完成时存储执行历史字段中的信息。指令历史信息可以表示在指令组中的指令是否具有对另一个指令依赖性，或者可以表示先前的指令组的执行是否引起存储输送异常。在这个实施例中，处理器被构成用于响应检测到指令组先前的执行引起过存储输送异常而以有序方式执行指令。

本发明还提出一种利用第一发出队和第二发出队的处理器、数据处理系统和相关的方法。所述处理器适用于向发出单元分派指令。所述发出单元包括第一发出队和第二发出队。如果指令在当前不满足被发出的条件则被存储在第一发出队中。如果指令在当前满足被发出的条件则被存储在第二发出队中。处理器从第一发出队的指令中确定要发出的下一个指令。如果指令依赖于另一个指令的执行结果，则可以从第一发出队中转移到第二发出队中。在一个实施例中，指令可以在被发出以便执行之后被从第一发出队中转移到第二发出队中。在这个实施例中，在一个特定的时间间隔内，指令可以被保存在第二发出队中。此后，如果该指令尚未被拒绝，则包含

该指令的第二发出队入口不再被分配。所述微处理器包括指令高速缓存，分配单元，其被构成用于接收来自指令高速缓存的指令，以及发送单元，其被构成用于接收来自分配单元的指令。发送单元适用于分配当前够执行条件的分派的指令到第一发出队中，并把当前不够执行条件的指令分配到第二发出队中。

本发明的其它的目的和优点通过下面结合附图详细说明本发明可以更加清楚，其中：

图 1 是包括按照本发明的一个实施例的数据处理系统的选择的部分的方块图；

图 2 是按照本发明的一个实施例的微处理器的选择的部分的方块图；

图 3 表示由图 2 所示的处理器的一个实施例执行的指令分裂功能的一个例子；

图 4 是一种微处理器的选择的部分的方块图；

图 5 是图 2 的微处理器的基本高速缓存块的方块图；

图 6 说明图 2 的微处理器可能遇到的各种分支情况；图 7 是适用于利用本发明的一个填空表的方块图；

图 8 是包括指令历史信息的基本块高速缓存的方块图；以及

图 9 是包括按照本发明的一个实施例的一次发出队和二次发出队的发出队的方块图。

虽然本发明可以用各种不同的形式实施，此处还是以举例的方式在附图中示出了一些特定的实施例并加以详细说明。不过，应当理解，这里的附图和详细说明并不是为了把本发明限制于这些特定的实施例，而恰恰相反，本发明应当包括落在由所附权利要求限定的本发明的构思范围内的所有改型和等效替代方案。

参见图 1，其中示出了按照本发明的数据处理系统 100 的一个实施例。系统 100 包括一个或几个中央处理单元（处理器）101a, 101b, 101c 等（一般被称为处理器 101）。在一个实施例中，每个处理器 101 可以包括简化指令系统计算机（RISC）微处理器。关

于 RISC 处理器的附加信息一般可以在 C.May et al.Ed.,PowerPC Architecture:A Specification for a New Family of RISC Peocessors (Morgan Kaufmann,1994 2d edition) 得到。处理器 101 通过系统总线 113 和系统存储器 250 以及各种其它元件相连。只读存储器 (ROM) 102 和系统总线 113 相连, 并且可以包括基本输入输出系统 (BIOS), 其用于控制系统 100 的某个基本功能。图 1 还示出了 I/O 适配器 107 和网络适配器 106, 它们和系统总线 113 相连。I/O 适配器 107 连接系统总线 113 和大容量的存储装置 104 例如硬盘 103 和/或磁带存储驱动器 105。网络适配器 106 互连总线 113 和外部网络, 以便使数据处理系统 100 能够和其它这样的的系统进行通信。显示监视器 136 通过显示适配器 112 和系统总线 113 相连, 其中可以包括图形适配器, 用于改善图形密集应用的性能, 以及视频控制器。在一个实施例中, 适配器 107, 106 和 112 可以通过中间总线桥 (未示出) 和一个或几个与系统总线 113 相连的 I/O 总线。适用于连接外围设备例如硬盘控制器、网络适配器、和图形适配器的 I/O 总线包括按照 PCI 本地总线规范 Rev.2.2 规定的外围元件接口 (PCI) 总线, 其可以从 PCI Special Interest Group,Hillsboro,OR, 得到, 在此列为参考。还示出了通过用户接口适配器 108 和系统总线 113 相连的附加的输入输出装置。键盘 109、鼠标 110、以及扬声器 111 都通过用户接口适配器 108 和总线 113 相连, 所述用户接口适配器可以包括把多个设备适配器集成一个集成电路的超 I/O 芯片。关于这种芯片的其它信息, 读者可以参阅 PC87338/PC97338 ACPI 1.0 and PC98/99 Compliant SuperI/O data sheet from National Semiconductor Corporation (November 1998) at 微处理器.national.com.. 这样, 如图 1 所示的配置, 系统 100 包括呈处理器 101 的形式的处理装置, 包括系统存储器 250 和大容量存储器 104 的存储装置, 输入装置例如键盘 109 和鼠标 110, 以及包括扬声器 111 和显示器 136 的输出装置。在一个实施例中, 系统存储器 250 和大容量存储器 104 集中存储一种操作系统例如 IBM 公司的 AIX®操

作系统或其它合适的操作系统，用于协调图 1 所示的各种元件的功能。关于 AIX 操作系统的其它细节可以在 AIX Version 4.3 Technical Reference: Base Operating System and Extensions, Volumes 1 and 2 (order numbers SC23-4159 and SC23-4160); AIX Version 4.3 System User's Guide: Communications and Networks (order number SC23-4122); and AIX Version 4.3 System User's Guide: Operating System and Devices (order number SC23-4121) from IBM Corporation at 微处理器.ibm.com 得到，在此列为参考。

参见图 2，其中示出了按照本发明的一个实施例的处理器 101 的一个简化的方块图。图 2 所示的处理器 101 包括适用于产生要被读取的下一个指令的地址的指令读取单元 202。由指令读取单元 202 产生的指令地址被提供给指令高速缓存 210。读取单元 202 可以包括分支预测逻辑，顾名思义，其用于进行关于影响程序执行流程的决定结果的预测。为了通过推测地和无序地执行指令来实现改善性能，在处理器 101 的所有的能力中，正确地预测分支决定的能力是一个重要的元素。由读取单元 202 产生的地址被提供给指令高速缓存 210，其在一种高速存储装置中包含一个系统存储器内容的子集。在指令高速缓存 210 中存储的指令最好具有第一 ISA 的格式，其一般是一种传统的 ISA，例如 PowerPC 或 x86 兼容的指令集。关于 PowerPC® 指令集的详细信息可以在 PowerPC 620 RISC Microprocessor User's Manual available from Motorola, Inc. (Order No.MPC620UM/AD) 中得到，在此列为参考。如果由读取单元 202 产生的地址指令相应于当前被复制在指令高速缓存中的系统存储器位置，则指令高速缓存 210 把相应的指令送到指令分裂单元 212。如果相应于由读取单元 202 产生的指令地址的指令当前不在指令高速缓存 210 中（即由读取单元 202 提供的指令地址不在指令高速缓存 210 中），则在指令被送到分裂单元 212 之前，指令必须从 L2 高速缓存（未示出）或系统存储器中被读取。

分裂单元 212 适用于修改输入的指令流，从而产生一组优化的

指令，用于以高的操作频率（例如超过 1GHz 的操作频率）在下面的执行流水线中执行。在一个实施例中，例如，分裂单元 212 以 32 位宽 ISA 接收指令，例如由 PowerPC® 支持的指令组，并把指令转换成最好是较宽的第二 ISA，其易于在以 1GHz 或更高的频率范围内操作的高速执行单元中执行。由分裂单元 212 产生的指令的较宽的格式可以包括例如含有信息（例如操作数值）的显式字段，所述信息只在由分裂单元 212 接收的指令中隐含或参考，所述字段按照第一格式被格式化。在一个实施例中，例如，由分裂单元 212 产生的指令的 ISA 具有 64 位或更宽的宽度。

在一个实施例中，设计分裂单元 212，如此处所设想的，除去把指令从第一格式转换成最好是较宽的第二格式之外，还把读取的指令集组织在指令“组”302 中，其例子如图 3 所示。每个指令组 302 包括一组指令位置 304a, 304b 等（一般被称为指令位置 304）。把指令集组织在指令组中有利于高速执行，其中主要通过对于大量的飞行中的指令简化为保持重新命名寄存器变换和完成表所需的逻辑。在图 3 中，示出了通过分裂单元 212 可以实现的指令组的 3 个例子。

在例 1 中，由标号 301 表示的一组指令由分裂单元 212 转换成一个单个的指令组 302。在所示的本发明的实施例中，每个指令组 302 包括由标号 304a, 304b, 304c, 304d, 304e 表示的 5 个位置。每个位置 304 可以包含一个指令。在本实施例中，每个指令组最多可以包括 5 个指令。在一个实施例中，由分裂单元 212 接收的指令集 301 中的指令按照第一 ISA 被格式化，如上所述，并且存储在组 302 中的指令按照第二较宽的格式被格式化。使用指令组通过减少必须被单独启动和跟踪的指令的数量简化了重新命名恢复的和完成的表逻辑。因而，在无序处理器中，估计使用指令组在力图简化跟踪挂起指令时会牺牲关于每个指令的一些信息。

图 3 的例 2 说明按照本发明的一个实施例由分裂单元 212 实现的指令组的第二个例子。这个例子表示分裂单元 212 把复杂的指令组织成简单的指令组以便高速执行的能力。在所示的例子中，两个更

新加载 (LDU) 指令序列被组织成指令组, 所述指令组包括分别位于 304a 和 304c 的一对加载指令和分别位于 304b 和 304d 的一对 ADD 指令。在这个例子中, 因为组 302 不含有分支指令, 所以指令组 302 的最后位置 304e 不含有指令。PowerPC®更新加载指令和在其它指令集中的类似指令一样, 是一种复杂的指令, 其影响多个通用寄存器 (GPR) 的内容。具体地说, 更新加载指令可以被分成影响第一 GPR 的内容的加载指令和影响第二 GPR 的内容的 ADD 指令。因而, 在图 3 的例 2 的指令组 302 中, 两个或多个指令位置 304 中的指令相应于由分裂单元 212 接收的一个指令。

在例 3 中, 输入到分裂单元 212 的单个指令被分成占据多个组 302 的一组指令。更具体地说, 例 3 说明加载多重 (LM) 指令。加载多重指令 (按照 PowerPC®指令集) 将存储器中的连续的位置的内容装入被连续编号的 GPR 中。在所示的例子中, 6 个连续的存储器位置的多次加载被分成 6 个加载指令。因为按照所述的处理器 101 的实施例的每个组 32 最多包括 5 个指令, 并且因为第 5 个位置 304e 被保留用于分支指令, 所以 6 个寄存器的多次加载分别被分成两组 302a 和 302b。加载指令当中的 4 个被存储在第一组 302a 中, 而其余的两个加载指令被存储在第二组 302b 中。因而, 在例 3 中, 一个指令被分成生成多个指令组的指令集。

现在参看图 2, 由分裂单元 212 的优选实施例产生的指令组 302 被送到基本块高速缓存 213, 在那里它们被存储等待执行。参看图 5, 其中输出量基本块高速缓存 213 的实施例。在所示的实施例中, 基本块高速缓存 213 包括一组入口 502a-502n (一般被称为基本块高速缓存入口)。在一个实施例中, 在基本块高速缓存 213 中的每个入口含有一个指令组 302。此外, 每个入口 502 可以包括入口识别符 504, 指针 506 和指令地址 (IA) 字段 507。每个入口 502 的指令地址字段 507 类似于完整表 218 的 IA 字段 704。在一个实施例中, 在基本块高速缓存 504 中的每个入口 502 相应于在完整表 218 中的入口, 并且指令地址字段 507 表示在相应的指令组 302 中的第一指令

的指令地址。在一个实施例中，指针 506 根据分支预测算法、分支历史表或其它合适的分支因此机构表示要被执行的下一个指令组 302 的入口识别符。如前所述，利用分裂单元 212 形成指令组的优选的实施方案在每个指令组 302 中的最后位置 304 分配分支指令。此外，分裂单元 212 的优选实施例产生指令组 302，其中在组 302 中的分支指令的数量是 1（或更少）。在这种结构中，每个指令组 302 可以被看作代表图 6 所示的分支树 600 的一条“腿”，其中指令组 302 用其相应的指令组入口 504 的值表示。例如第一指令组 302a 用入口数（1）表示，依此类推。作为例子，假定处理器 101 的分支预测机构预测腿 2（相应于第二组 302b）将在腿 1 之后被执行，并且腿 3 将在腿 2 之后被执行。按照本发明的一个实施例，基本块高速缓存 213 通过设置指针 506 指示要被执行的下一组 302 来反映这些分支预测。在基本块高速缓存 213 中的每个入口 502 的指针 506 可被用于确定要被调遣的下一个指令组 302。

基本块高速缓存 213 和块读取单元 215 以读取单元 202 和指令高速缓存 210 协同操作的的方式类似的方式协同操作。更具体地说，块读取单元 215 负责产生被提供给基本块高速缓存 213 的指令地址。被块读取单元 215 提供的指令地址和基本块高速缓存 213 中的指令地址字段 507 中的地址相比较。如果被块读取单元 213 提供的指令地址在基本块高速缓存 213 中，则向发出队 230 输入合适的指令组。如果由块读取单元 215 提供的地址不在基本块高速缓存 213 内，则指令地址被回送到读取单元 202，从指令高速缓存 210 中检索合适的指令。在用于其区域（小片尺寸）保存的一个实施例中，基本块高速缓存 213 能够取消指令高速缓存 210。在这个实施例中，指令从一个合适的存储装置例如 L2 高速缓存或系统存储器中被检索，并被直接提供给分裂单元 212。如果由块读取单元 213 产生的指令地址不在基本块高速缓存 213 中，则从 L2 高速缓存或系统存储器中而不从指令高速缓存 210 中检索合适的指令。

所示的处理器 101 的实施例还表示调度单元 214。调度单元 214

负责确保在在对其合适的发出队 220 输入每个指令组中的指令之前所有所需的资源都是可得到的。此外，调度单元 214 与调度和完成控制逻辑 216 通信，以便保持跟踪指令被发出的顺序和这些指令的完成状态，以利于无序执行。在处理器 101 的实施例中，其中分裂单元 212 把输入指令组织成为指令组，如上所述，每个指令组 302 通过传递发出的指令组的顺序的完成和控制逻辑 216 分配一个组标记（GTAG）。作为一个例子，调度单元 214 可以对连续的指令组分配一个单调增加的值。利用这种结构，使得知道具有较低的 GTAG 值的（即较新的）指令组已经在具有较大的 GTAG 值的指令组之前被发出。虽然处理器 101 的所示的实施例表示调度单元 214 是一个单独的功能块，但是基本块高速缓存 213 的组指令组织使其自身包括了调度单元 214 的功能。因而，在一个实施例中，调度单元 214 被包括在基本块高速缓存 213 内，其和发出队 220 直接相连。

和调度与完成控制逻辑 216 相结合，完成表 218 在本发明的一个实施例中被用于跟踪发出的指令组的状态。参看图 7，其中示出了完成表 218 的一个实施例的方块图。在所示的实施例中，完成表 218 包括一组入口 702a-702n（被称为完成表入口 702）。在这个实施例中，在完成表 218 中的每个入口 702 包括指令地址（IA）字段 704 和状态位字段 706。在这个实施例中，每个指令组 302 的 GTAG 值识别完成表 218 中的入口 702，在完成表 218 中存储着相应于指令组 302 的完成信息。因而，在完成表 118 的入口 1 中存储的指令组 302 的 GTAG 的值为 1，依此类推。在这个实施例中，完成表 118 还可以包括“回绕”位，用于表示具有较低的 GTAG 值的指令组实际上比具有较高的 GTAG 值的指令组新。在一个实施例中，指令地址字段 704 包括相应的指令组 302 的第一位置 304a 中的指令的地址。状态字段 706 可以含有一个或几个状态位，用于指示例如在完成表 218 中的相应的入口 702 是否是可以利用的，或者该入口已经被分配给一个待执行的指令组。

在图 2 所示的处理器 101 的实施例中，指令从调度单元 214 向发

出队 220 被发出，指令在发出队等待在相应的执行管道中被执行。处理器 101 可以包括各种类型的执行管道，每个管道被指定用于执行处理器的指令集的一个子集。在一个实施例中，执行管道 222 可以包括分支单元流水线 224，负载存储流水线 226，固定点算法单元 228，和浮动点单元 230。每个执行管道 222 可以包括两个或多个流水线级。在发出队 220 中存储的指令可以通过使用任何一个发出优先权算法被发送给执行管道 222。在一个实施例中，例如在发出队 220 中的最早的待执行指令是被发送给执行管道 222 的下一个指令。在这个实施例中，由调度单元 214 指定的 GTAG 值被用于确定在发出队 220 中待执行的指令的相关的级。在被发出之前，指令的目标寄存器操作数被指定给可利用的重新命名 GPR。当一个指令最后从发出队 120 发送到合适的执行管道时，该执行管道则按照该指令的操作码的指示进行合适的操作，并在该指令到达流水线的最后一级时(由标号 132 表示)把执行结果写入指令的重新命名 GPR 中。在重新命名的 GPR 和其相应的结构寄存器之间保持一种映射。当在指令组中的所有的指令(以及在较近的指令组中的指令)都被完成而没有产生例外时，在完成表 218 中的完成指针被递增到下一个指令组。当完成指针递增到新的指令组时，和老指令组中的指令相关的重新命名寄存器被释放，借以用于寄存在老的指令组中的指令的结果。如果一个或几个比完成的指令老的指令(但是尚未被提交)产生了一个异常，则产生异常的指令和所有较新的指令都被刷新，并调用重新命名恢复程序，使 GPR 变换返回最近得知的有效状态。

如果预测的分支未被得到(分支预测错误)，则在指令管道 222 和发出队 220 中等待执行的指令被刷新。此外，和错误预测的分支相关的基本块高速缓存入口 502 的指针 506 被更新，从而反映最近得到的分支。图 5 示出了这种更新处理的一个例子，所示的情况是，程序的执行产生从腿 1(指令组 302a)到腿 4(指令组 302d)的一个分支。因为入口 502a 的指针 506 以前预测过到基本块高速缓存 213 的 2 号入口中存在的指令组(即组 302b)的分支，所以从指令组 302a

到组 302d 的实际的分支是误预测的。误预测的分支被检测，并被送回块读取单元 215，在基本块高速缓存 213 和每个流水线 222 的最后级 232 之间悬挂着的指令被刷新，并且利用在基本块高速缓存 213 的入口 4 中的指令组 302d 重新开始执行。此外，基本块高速缓存入口 502a 的指针 506 被从其以前的值 2 改变到新的值 4，以便反映最新的分支信息。通过在极接近于执行流水线 222 处合并基本块高速缓存 213 和块读取单元 215，本发明预计能够减少误预测分支的性能损失。更具体地说，通过在指令分裂单元 212 的“下游侧”上实施基本块高速缓存 213，本发明取消了在分裂单元 212 中悬挂的来自分支误预测刷新通路的指令，从而减少了由于分支误预测而必须清洗的流水线的级数，因而减少了性能损失。此外，基本块高速缓存 213 包括一个高速缓冲存储机构，其具有和调度与完成控制单元 216 以及完成表 218 的组织相匹配的结构，借以简化插入逻辑的组织，并帮助实现对基本块高速缓存 213 的有用的扩展，如下所述。

在一个实施例中，基本块高速缓存 213 还包括指令历史信息，这能够借助于记录在同一指令组的随后的执行期间可能使用的信息，使得避免可能引起异常、刷新、中断或其它性能限制事件（被统称为异常事件）的情况，从而改善处理器的性能。在图 8 所示的基本块高速缓存 213 的实施例中，指令历史信息被存储在每个入口 502 的指令历史字段 508 中。作为可以被存储在指令历史字段 508 中的信息类型的一个例子，可以举出包含一个特定的加载指令的指令组，所述指令组在所述加载指令最后一次被执行时导致存储前面的异常。此处所用的术语“存储前面的异常”当在无序计算机中存储指令之前执行在共用公共存储器参考的存储指令之后（按照程序顺序）的加载指令时发生。因为如果在存储指令之前执行负载指令，负载指令则从寄存器中检索无效的值，所以产生一个引起指令刷新的异常。在基本块高速缓存 213 和完成与控制逻辑 216 之间的对应性大大有助于完成向基本块高速缓存 213 的相应的入口输送由调度和完成控制逻辑 216 以指令被执行和完成的方式得知的信息的任

务。在没有所述对应性的情况下，来自调度和完成控制逻辑 216 的完成信息一般要求以插入散列表或其它合适的机构的一些方式通过，以便使组指令信息和其元件指令相关。在存储输送的例子中，在检测到存储输送异常时，调度和完成控制单元 216 将把合适的入口的指令历史字段 508 中的一位或几位写在应当表示存储输送异常的基本块高速缓存 213 中。如果指令组被随后执行，则可以使用表示以前发生的存储输送异常的指令历史信息，例如，为了把处理器置于有序方式，其中在完成存储之前阻止执行加载。因而，本发明的这一实施例考虑记录表示和指令组相关的异常事件的指令历史信息，并且此后修改指令组的执行，以便在随后的指令组的执行期间阻止异常事件的发生。虽然以存储输送为例进行了说明，但是此处考虑的指令历史信息字段也适合于记录和各种可能使处理器避免重新发生异常状态的历史事件相关的信息，例如和任何预测机构的精度、操作数值的预测、高速缓存的有无等相关的信息。

可以在基本块高速缓存 213 的执行历史字段 508 中被记录的信息的一个例子由图 9 所示的实施例说明，其中一个或几个发出队 220 被细分为第一发出队 902 和第二发出队 904。发出队 220 的最佳的尺寸和深度表示完成考虑之间的平衡。在一方面，希望执行非常大和非常深的发出队，以便最大限度地利用处理器 101 的执行无序指令的能力。发出无序指令的能力受到在发出队 220 中的指令数量的限制。较大的数量的发出队相应于较大数量的适合于无序处理的指令。在另一方面，随着发出队 220 的深度的增加，处理器在其周期时间限制内确定下一个被发出的指令的能力也增加。换句话说，在发出队 220 中悬挂的指令的数量越大，用于确定下一个要发出的指令所需的时间也越长。因此，发出队例如发出队 220 通常被限制于大约 20 或更少的深度。本发明的一个实施例考虑利用深的发出队而不需过大地增加检索下一个能够发出的指令的发出队所需的逻辑的优点。本发明利用这样的事实，即，通常在发出队 220 中悬挂的指令不能被立即发出，这或者是因为其已经被发出，或者是因为其被

悬挂在处理器 101 的执行流水线 222 中，或者其正在等待因操作数值而所依赖的另一个指令的完成。

参看图 9，按照本发明的一个实施例的发出队 220 包括第一发出队 902 和第二发出队 904。第一发出队 902 含有能够被立即发出的指令。在一个实施例中，从调度单元 214 中派遣的指令首先被存储在发出队 902 的可得到的发出中。如果随后确定一个指令具有和另一个指令的相关性，则相关的指令被移动到第二发出队 904，直到相关的指令所依赖的指令检索到所需的信息。例如，如果在加载指令之后的一个附加的指令需要加载指令的结果，则可以把两个指令首先调入第一发出队 902 中。不过，在接着确定附加的指令具有对加载指令的依赖性时，附加的指令被从第一发出队 902 被转移到第二发出队 904。在上面参照图 8 所述的利用指令历史字段 508 的实施例中，可以记录附加指令的依赖性，使得在该指令的随后执行期间，附加指令可被直接存储在第二发出队 220 中。也可以使用第二发出队 904 存储最近被发出的仍然被悬挂在处理器的执行流水线中的指令。在本实施例中，指令从第一发出队 902 发出，然后被转移到第二发出队 904。在一个实施例中，指令可以驻留在第二发出队 904 中，直到确定该指令不被拒绝为止。确定一个指令尚未被拒绝的一个方法是实施一个和第二发出队 904 中的每个发出相关的计时器/计数器（未示出）。当开始把指令从第一发出队 902 转移到第二发出队 904 时，计数器/计时器被启动。在一个实施例中，计时器/计数器对从计时器/计数器被启动时已经经过的时钟周期的数量计数。如果计数器/计时器在一个预定的周期数之内继续计数，则没有检测到指令已被拒绝，指令被假定已经成功地被完成，并且在第二发出队 904 中的项不再被分配。通过利用这样的一个发出队，其中包括第一发出队和第二发出队，所述第一发出队专用于当前够条件发出以便被执行的指令，在所述第二发出队中的指令或者因为指令的相关性，或者因为该指令是最近从第一发出队发出的而尚未被发送，但是所述指令当前不够执行条件，可以增加发出队的有效尺寸或深度，而

不显著地增加用于确定下一个要被发出的指令所需的时间（即逻辑电平的数量）。

本领域的技术人员应当理解，根据上面披露的内容，可以设想许多不同的微处理器的实施例，其中包括适合于存储分组指令（即已被从第一格式转换为第二格式的指令）的高速缓存，以便减少误预测的分支有关的等待时间。应当理解此处详细说明的本发明的形式仅仅是目前的优选的例子，旨在在一个宽的范围内解释下面的权利要求，所附的权利要求包括这些实施例的所有的改变和改型。

说明书附图

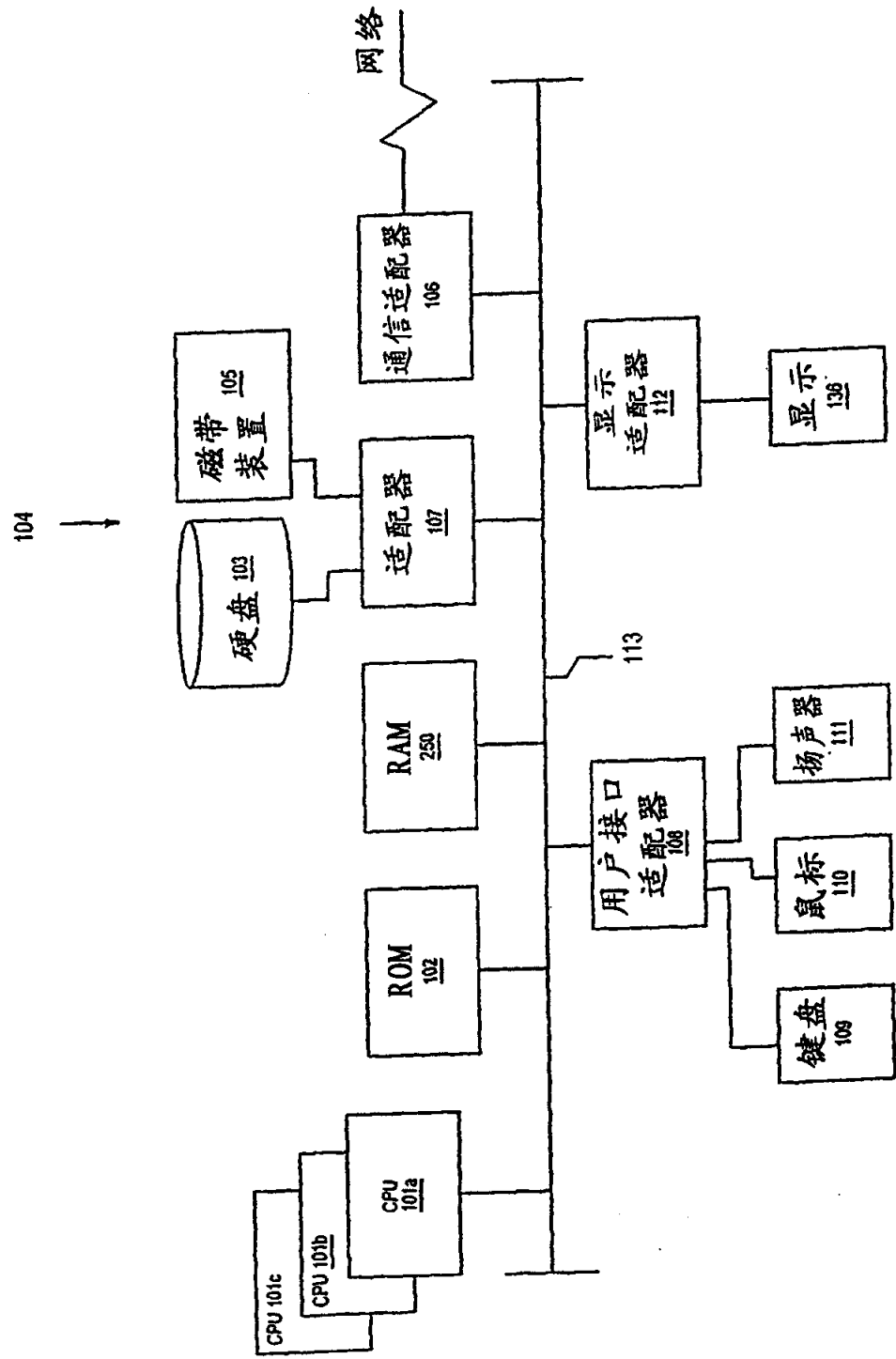


图1

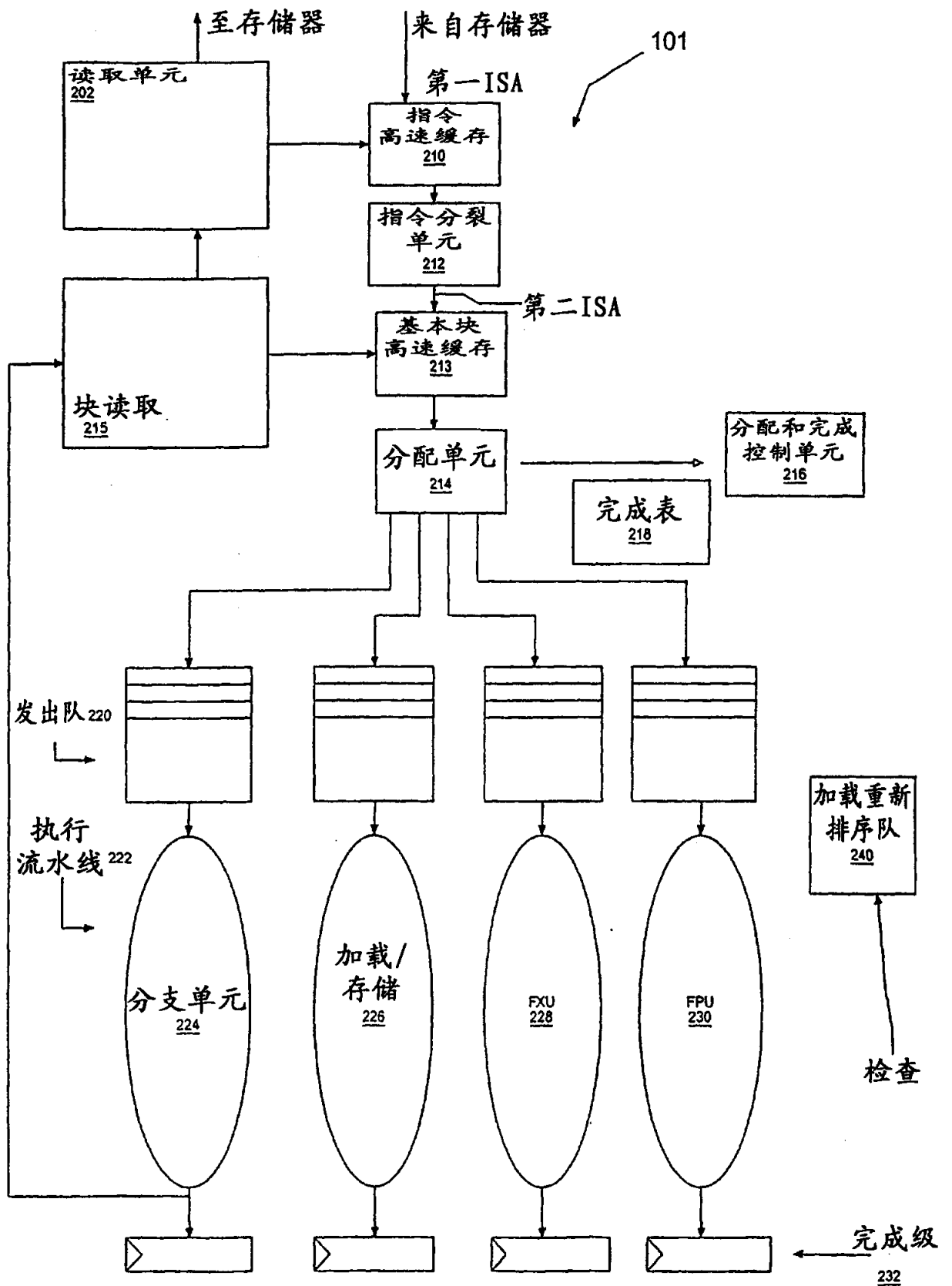
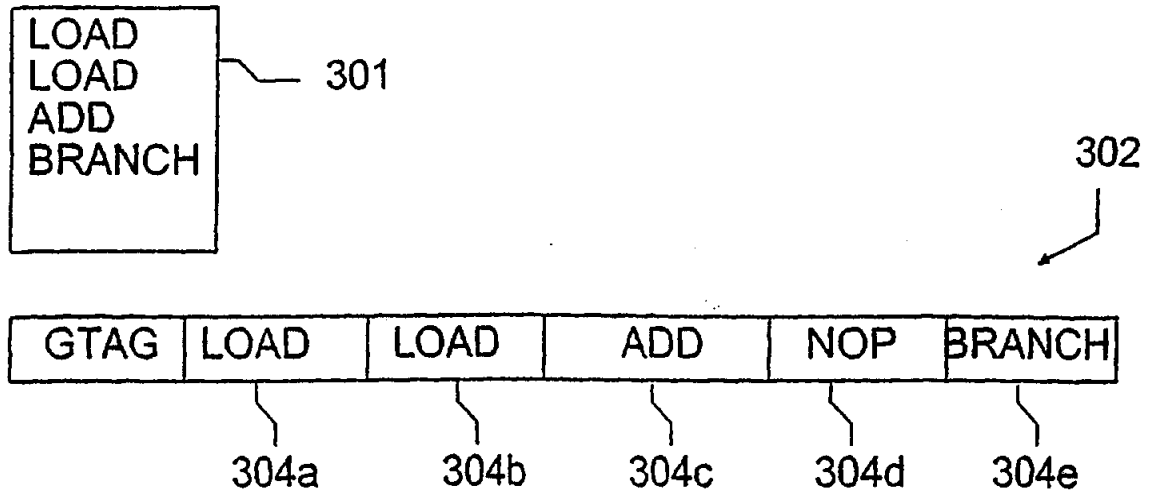
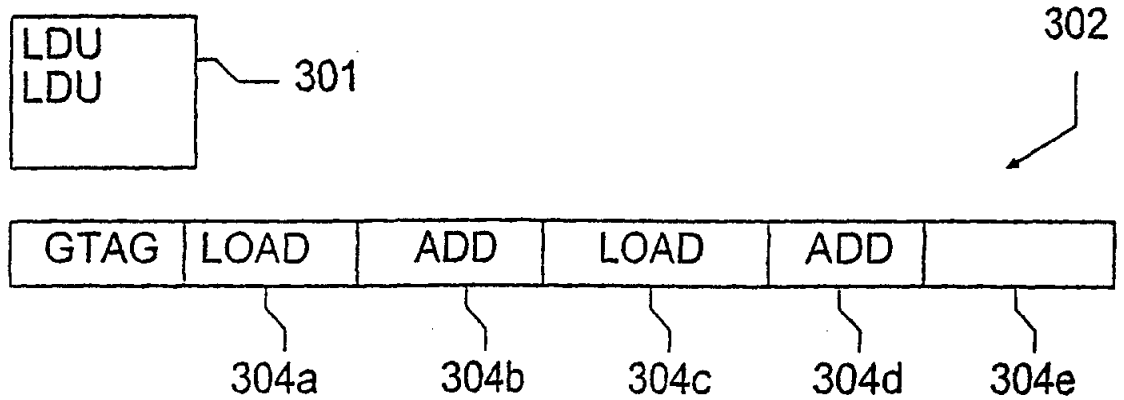


图 2

例1



例2



例3

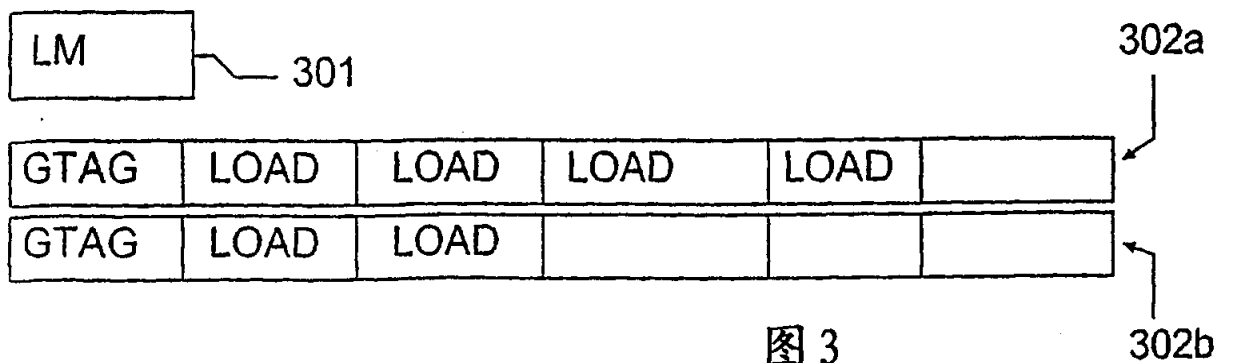


图 3

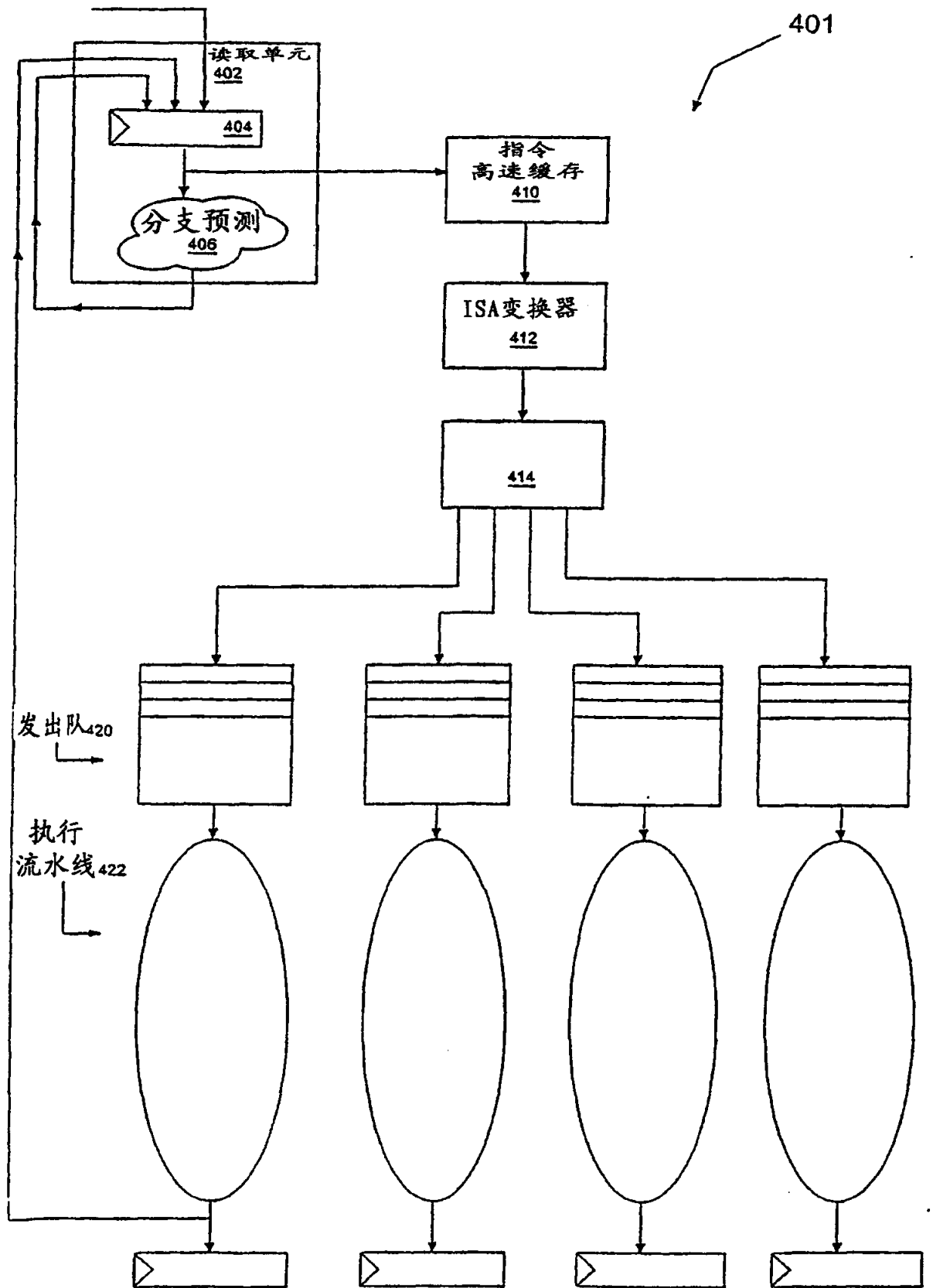
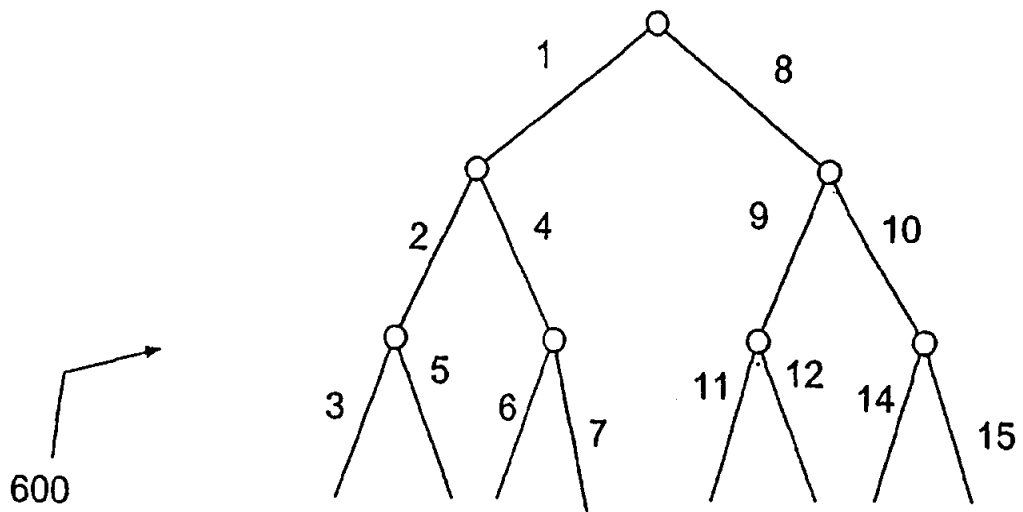
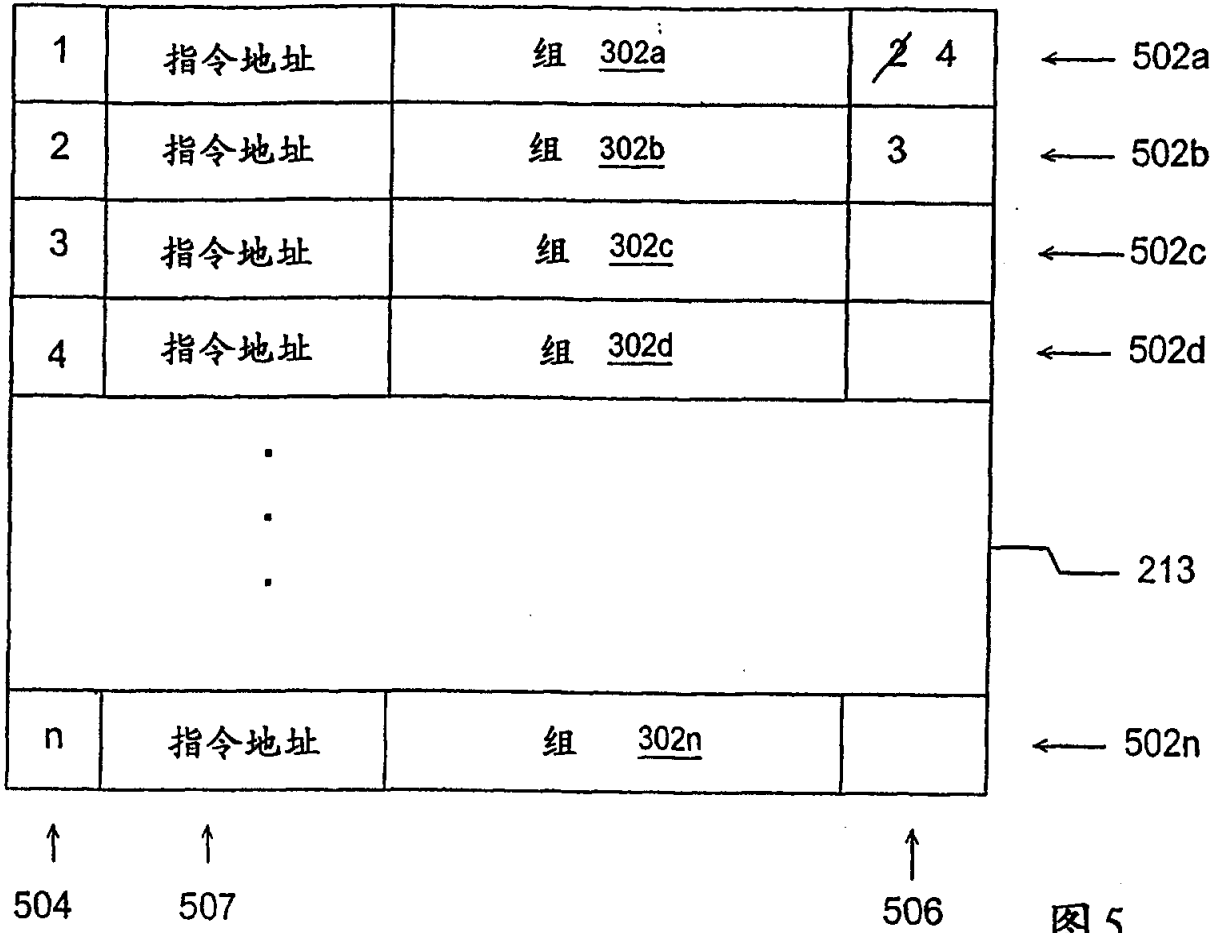


图 4



回绕位 703

	指令地址 <u>704</u>	状态 <u>706</u>
702a		
702n		

218

图 7

1	指令历史	指令地址	组 <u>302a</u>	4	← 502a
2	指令历史	指令地址	组 <u>302b</u>	3	← 502b
3	指令历史	指令地址	组 <u>302c</u>		← 502c
4	指令历史	指令地址	组 <u>302d</u>		← 502d
. . .					← 213
n	指令历史	指令地址	组 <u>302n</u>		← 502n

↑
504

↑
508

↑
507

↑
506

图 8

