[19]中华人民共和国专利局

[11] 公开号 CN 1127899A

[12] 发明专利申请公开说明书

[21]申请号 94119533.3

[51]Int.Cl[6]

G06F 9/06

[43]公开日 1996 年 7 月 31 日

[22]申请日 94.12.17

[30]优先权

[32]93.12.20[33]US[31]169262

[71]申请人 摩托罗拉公司

地址 美国伊利诺斯

[72]发明人 丹恩·K·杰伊 戴维·S·莱维塔 保罗·C·罗斯巴赫

[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

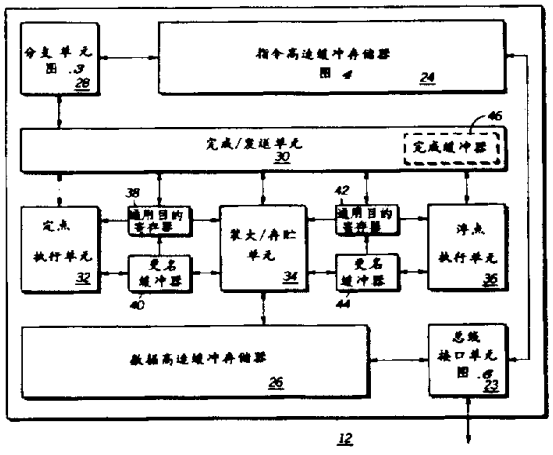
代理人 范本国

权利要求书 3 页 说明书 24 页 附图页数 5 页

[54]发明名称 具有推测指令取指的数据处理器及操作方法

[57]摘要

数据处理器（12）具有预测条件分支指令的分支预测单元（28）和监控未决定分支指令数的控制单元（70），控制单元根据未决定分支指令数有选择地允许数据处理器从外部存储器系统取由分支预测单元指定的指令，未决定分支指令的特点阈值数可由用户编程，数据处理器从而把其总线存取限制于那些可以确定需要指示指令的场合。



权 利 要 求 书

1. 利用猜测指令取指的数据处理器，它使用外部存储系统，这种数据处理器其特征在于

由分支预测单元形成取指地址，由分支预测单元存储一信号用以表示未决定分支指令数，由分支预测单元产生一控制信号，在第一种操作方式中若信号表示数量分别小于等于第一预选择数或大于第一预选择数时控制信号分别对应于第一逻辑状态及第二逻辑状态，在第二种操作方式中，若信号表示数分别小于等于第二预选择数或大于第二预选择数时，控制信号分别对应于第一逻辑状态及第二逻辑状态；

指令取指电路连接到分支预测单元，指令取指电路取出一指令用来自外部存储器系统对应于控制信号的第一状态的取指地址变址。

2. 按权利要求 1 的数据处理器中分支预测单元其特征在于：

分支计算电路形成与分支指令对应的第一和第二地址；

分支预测电路连到分支计算电路，分支预测电路输出取指地址取指地址对应于条件逻辑上等于第一或第二地址中选择的一个。

分支完成电路用来存贮表示未决定分支指令数的信号。

3. 按权利要求 2 的数据处理器中分支完成电路其特征在于电

路用来存贮逻辑上不等于取指地址的第一和第二地址中一个。

4. 按权利要求 3 的数据处理器其特征在于用高速缓存的存储器连到指令取指电路，高速缓存存储器装置存贮从外部存储系统接收的指令。

5. 按权利要求 1 的数据处理器其特征在于高速缓存存储器连到指令取指电路，高速缓存存储器装置存贮从外部存储系统接收的指令。

6. 带有外部存储器系统的数据处理器的工作方法其特征在于下面步骤：

接收，在数据处理器的分支预测单元中接收分支指令；

预测，在分支预测单元中对应于分支指令预测一取指地址；

存贮，在分支预测单元中，存贮一未决定分支指令的数目；

生成，在分支预测单元中，形成控制信号，在第一种操作方式中，若未决定分支指令数分别小于等于第一预选数或大于第一预选数时，控制信号分别对应于第一种逻辑状态和第二种逻辑状态，在第二种操作方式中，若未决定分支指令数分别小于等于第二预选数或大于第二预选数时，控制信号分别对应于第一种逻辑状态和第二种逻辑状态；

第一取指，用指令取指电路连到分支预测单元，根据控制信号从外部存储器系统取出指令。

7. 按权利要求 6 的方法中预测步骤其特征在于下面步骤：

生成，在分支计算电路中，根据分支指令，生成第一和第二地址；输出，用分支预测电路连到分支计算电路，输出的取指地址逻辑上等于所选择的第一或第二地址之一。

8. 按照权利要求 7 的方法中存贮步骤其特征在于存贮的第一或第二地址逻辑上不等于取指地址。

9. 按照权利要求 8 的方法其特征在于第二取指步在第一取指步之前，第二取指步经指令取指电路，从数据处理器的高速缓存存储器取指。

10. 按照权利要求 6 的方法其特征在于第二取指步在第一取指步之前，第二取指步经指令取指电路，从数据处理器的高速缓存存储器取指。

说　　明　　书

具有推测指令取指的数据处理器及操作方法

本发明一般性地涉及数字计算系统，更特别涉及具有猜测指令取指能力的数据处理器。

分支预测是一种用来改进数据处理器性能的技术，用分支预测技术的数据处理器，每当它们接收一分支指令时总要进行猜测，实行猜测（未决定分支指令），并通过完成指令（决定分支指令）确定猜测是否正确。这种数据处理器猜测是否最终实现分支并转到新的指令地址（不是往下到另一顺序的指令。具有预测分支指令的数据处理器，因为它们形成正确猜测比完成分支指令更快而增加了其性能。这样这些数据处理器所需的仅仅是校正错误猜测。

分支预测技术允许数据处理器尽可能快的在取下一个指令或指令组而增加数据处理器性能。因此，数据处理器总有连续指令流在执行。这种策略称“指令预取指”，因为数据处理器在它执行分支指令之前取指令以确定预取指令的地址。

在某些数据处理系统中，统一的取指指令，对系统的整个性能可能不利。通常，在数据处理器的内部存储器的高速缓存中不包括

预取指指令。在这种情况下，数据处理器必须从外部存储器系统取指。这种操作垄断同外部存储装置相连的总线，同时可使共用外部存储器装置的其它装置实现同系统存储器有关协议（探 听操作）相符合的某些作用。另外，数据处理器直到接收居先的不正确指令才能请求其它的、可假定是正确的指令。其优点是只要可能，就使这些操作的次数变为最小。

每个猜测的分支指令对所选的指令通道混有不正确的东西，因此可能 有不需要的取指，例如有些分支预测方案，预测分支正确率可达 90％不正确率为 10％。经三次预测仍未决定分支，数据处理器沿正确指令流通道进行取指的机遇只有 73％。继续用这例子，已知数据处理器同样猜测第一和第 3 分支预测，既可通过两个分支指令指示的地址（或者是在两个指令之后形成的顺序地址）预取指令亦可不预取指令，如上所述，若数据处理器不包括由分支指令返回的特殊指令，这两指令中每个都可以使数据处理器经系统总线存取外部存储器系统。其它相同时，宁可垄断系统总线取第一指令而不取第 3 指令。然而，以前数据处理器没有这种特性。

根据本发明公布的具有推测指令取指的数据处理器，它实质上消除了已有的数据处理器缺点。

所用具有外部存储系统的数据处理器，具有分支预测单元及与 · 其相结合的指令取指电路。分支预测单元形成取指地址，存储一个表示前面未决定分支指令数目的信号及形成控制信号。在第一种操

作方式中，控制信号所对应的第一种逻辑状态和第二种逻辑状态分别表示信号代表数小于等于第一预选择数或大于第一预选择数。在第二种操作方式中，控制信号对应的第一种逻辑状态和第二种逻辑状态分别表示信号代表数小于等于第二预选择数或大于第二预选择数。指令取指电路从外部存储器系统与控制信号的第一状态相对应的取指地址间址取指。

用外部存储器系统的数据处理器的操作方法亦被说明，这种方法有在分支预测单元中有如下步，接收分支指令，对应于分支指令预测取指地址，存储未决定的分支指令数并形成控制信号。在第一种操作方式中，控制信号对应的第一种和第二种逻辑状态，分别对应于未决定的分支指令数小于等于第一预选择数及大于第一预选择数，在第二种操作方式中，控制信号对应的第一种和第二种逻辑状态分别对应于未决定的分支指令数小于等于第二预选择数及大于第二预选择数。这种方法还包括在同分支预测单元相联的指令取指电路中，对应于控制信号从外部存储系统中取指令的步骤。

通过结合附图的详细说明将会清楚的了解。本发明的特点和优点这些图号对应的部件如下：

图 1 根据本发明描绘了数据处理系统的框图结构。

图 2 描绘了在图 1 中描绘的数据处理器的框图。

图 3 描绘了在图 2 中描绘的分支单元的方框图。

图 4 描绘了在图 2 中描绘的指令高速缓存的框图。

图 5 描绘了在图 4 中描绘的指令高速缓存逻辑单元的状态转换图。

图 6 描绘了在图 2 中描绘的总线接口单元的框图。

图 1 描绘了根据本发明构成的数据处理系统 10 的方框图。数据处理系统 10 由第一数据处理器块 12，第二数据处理器块 14，主存储器块 16，总线仲裁块 18 和输入输出（以下称 I/O）块 20 经地址总线和数据总线内部相连而成。如上所述第二数据处理块 14 有外部高速缓存块 22（典型的二次或“L2”高速缓冲）。数据处理器 12 和 14 预测分支指令，预取通过分支单元 28 的返回地址所指示的指令，分支单元 28 以任意次序对应接收到的分支指令及后续无任何分支的指令，形成取指地址。而且，数据处理器 12 和 14 根据用户可编程位选择预取指令。每个数据处理器预取指令数取决于特定取指地址形成时有多少未决定的分支指令存在。每个数据处理器因此限制了经地址和数据总线存取主存储器块 16 的次数，例如当不正确分支猜测有高概率时。

本发明形成的数据处理器在单处理器（未示出）和多处理器（MP）数据处理系统（示出）中性能良好，在单处理器环境下，一个数据处理器可以是唯一的主存储器用户。在这种情况下一个不正确取指仅推迟一个处理器取出正确指令，相反，在多处理器环境下，不正确的取指将推迟处理器之一取出正确指令并将阻止其他处理器使用公共总线。根据本发明的数据处理器可以编程使在每种数据处理

系统中工作在最佳情况。

图 1 描述的许多功能块在技术上已是熟知的。数据处理块 12 和 14，利用存贮在主存储块 16 的数据和通过 I/0 块 20 接收的数据执行存储在主存储器块 16 中的指令。I/0 块 20 提供从数据处理系统 10 到键盘，盘驱动器和电子网络等的接口，总线仲裁器 18 接收来自数据处理系统 10 的各块请求，以使数据和地址总线排它性使用，总线仲裁块 18 根据协议（与本发明无关）受理这些请求。数据处理器块 14 存储频繁使用的数据在高速缓冲块 22 中。

图 2 描述了在图 1 描述的数据处理器 12 的方框图。数据处理器 12 和 14 实质上是相似的。因此，下面优选实施例将详细对数据处理器 12 来说明，必要时，说明它与数据处理器 14 的不同。

继续看图 2，总线接口单元（以后称 BIU）23 控制在数据处理器 12 和数据处理系统 10 的其余部分间的数据流。BIU 23 连到指令高速缓存 24 再到数据高速缓冲 26。指令高速缓存 24 提供指令流到分支单元 28，再到完成/调度单元 30。完成/调度单元 30 进而把各个指令送到合适的执行单元。数据处理器 12 有定点执行单元 32，装载/存储执行单元 34 和浮点执行单元 36。定点执行单元 32 和装载/存储执行单元 34 将其结果读和写到通用目的结构寄存器堆 38（标成 GPRS 以后称 GPR 堆）和第一更各缓冲器 40，浮点执行单元 36 和装载/存储执行单元 34 读和写其结果到浮点结构寄存器堆 42（标成

FPRS 以后称 FPR 堆）和第二更名缓冲器 44。

数据处理器 12 的操作在技术上是熟知的，它没有用推断指令预取指方法。通常，BIU 23 同数据处理系统 10 通信，BIU 23 将结合图 6 详述。分支单元 18 决定在给定某些数据寄存器内容和程序步骤的情况下，什么编程指令序列是合适的。分支单元 18 结合图 3 在下面详述。分支单元 38 首先希望从指令高速缓存 24 中取指，若所请求的指令没有存在指令高速缓存 24 中，那么指令高速缓存 24 将发送一请求到 BIU 23，从主存 16 去取请求指令。指令高速缓存 24 将结合图 4 和图 5 在下面详述。完成/调度单元 30 对不同的执行单元 32，34 和 36 发各自的指令。每个执行单元实现一个或多个特殊指令类的指令。每个执行单元的特殊指令类用执行单元的名字表示，如浮点执行单元 36 执行浮点算术指令。

定点执行单元 32 把操作的结果返回到第一更名缓冲器 40，中指定的条目项。当形成结果指令之前的所有指令已经更新了它们的 GPR 或 FDR 堆条目时，第一更名缓冲器 40 用它自己条目周期性地更新 GPR 堆 38 的条目。完成/调度单元 30 通过在完成缓冲器 46 中保留的所有执行指令表协调这种更新。第一更名缓冲器 40 和 GPR 堆 38 均对定点执行单元 32 提供操作数，相反，浮点执行单元 36 将运算结果返回第二更名缓冲器 44 中指定的条目中。当形成结果的指令之前的所有指令已经更新它们的 GPR 或 FPR 堆条目时，第二更名缓冲器 44 周期性用第二更名缓冲器 44 中条目更新 FPR 堆 42 的

条目。完成/调度单元 30 也协调这种更新，第二更名缓冲器 44 和 FPR 堆 42 均对浮点执行单元 36 提供操作数。

装载/存贮单元 34 读存于 GPR 堆 38，第一更名缓冲器 40，FPR 堆 42 或第二更名缓冲器 44 中的数据并把选定的数据写到数据高速缓存 26 中。这些数据亦可以通过同本发明无关的数据处理器 12 的操作特性写到主存储器系统 16 中。相反，装载/存储单元 34 读出存于数据高速缓存 26 的数据并把这些读出数据写到 GPR 堆 38，第一更名缓冲器 40，FPR 堆 42 或第二更名缓冲器 44。

公开的具有推测指令预取指方法的数据处理器 12 的操作下面，结合图 3 到图 6 说明，通常，数据处理器 12 是精简指令集计算机（“RISC”）。数据处理器 12 把每个指令断成一系列较小步，其中每步可以同其它指令在时间上重迭执行而得到高性能。这种操作策略称为“流水线技术”。数据处理器 12 亦可以通过把静态和动态分支预测方法相结合而得到高性能。数据处理器 12 可以用与用户编程值对应的静态或动态分支预测方式工作。而且，在用静态分支预测方法时。数据处理器 12 根据可编程位中一位，在分支历史表内更新或不更新每个分支指令的分支状态。

在所述的实施例中，每个指令断成 5 个分离步：取指，调度，执行，回写和完成。在指令高速 缓存 24 中存储器管理电路（未示出）在取指阶段时期，按分支单元 28 标识的存储器地址的起点取出一个或多个指令。完成/调度单元 30 在确定无不允许的数据从属，并在

调度阶段内对指令的结果保存更名缓冲器条目后发送每个指令到相应执行单元。每个执行单元在执行阶段执行程序指令并在回写阶段把结果（如果存在）写入被保留的更名缓冲器条目。最后，完成/调度单元 30，在特殊指令之前的每指令已经这样更新结构寄存器信息以后，用存储在更名缓冲器中特殊指令的结果更新结构寄存器信息。通常，每个指令相位取一个机器时钟周期，然而有些指令当其它指令不需要 5 个阶段时需要多于一个时钟周期去执行。由于完成各种指令的时间有不同，在特殊指令的回写和完成阶段之间也就可以出现延迟。

图 3 描绘了在图 2 中描绘的分支单元 28 的方框图、分支单元 28 形成下个指令地址以从存储器中取指（标记成 ADDRESS TO INSTRUCTION CACHE 24 指令高速缓存 24 地址）并从指令高速缓存 24（标记成 INSTRUCTIONS FROM INSTRUCTION CACHE 24 指令来自指令高速缓存 24）接收下个取出的指令。分支单元 28 也形成控制信号 GO TO BUS（到总线）和 OVERRIDE（取代），以确定是否数据处理器 12 存取地址和数据总线，以取未存储在指令缓冲器 24 中的指令。

分支单元 28 在调度缓冲器 48 中锁存接收的指令。在描绘实施例中，分支单元 28 的每个时钟周期接收 4 个指令：处在前一地址（取指地址）上的指令和跟在取指地址后三个地址上居留的 3 个指令。调度缓冲器 48 形成控制信号 DB EMPTY（DB 空）。调度缓冲器

48 当调度缓冲器 48 为空时置位 DB EMPTY。第一分支检测器 50 在 4 个锁存指令的每组内鉴别第一分支指令（如果存在）。第一分支检测器 50 形成控制信号 INCOMING BRANCH（引入分支）。第一分支检测器 50 当调度缓冲器 48 包含一个或多个分支指令时置位 INCOMING BRANCH 信号。

锁存于调度缓冲器 48 中的第一指令地址被传送到分支取指单元 52，分支调度/执行单元 54 及分支完成单元 56，分支取指单元 52，分支调度/执行单元 54 及分支完成单元 56 虽然处在不同的指令相位期，均形成一个地址，据此下一指令将被取出。分支调度/执行单元 54 也接收第一分支检测器 50 的输出。这三个单元的操作如下说明，多路开关选择器 58（标为 MUX）根据地址选择器 60 选择 3 个存储器地址之一输出。地址选择器 60 接收来自分支取指单元 52，分支调度/执行单元 54 和分支完成单元 56 的控制信号，将其译码确定 3 个存储地址的那个地址输出。指令取指地址寄存器 62（下文标成 IFAR）锁存多路开关选择器 58 的输出，IFAR62 形成标识为 ADDRESS TO MEMORY（到存储器的地址）的信号。

一旦 IFAR 62 锁存当前取指地址，分支取指单元 52 就为一特定分支指令计算 2 个新取指地址。这一计算发生在特定分支指令的取指阶段。由取指地址变址地特定指令在取指阶段并不能被知道。分支取指单元 52 输出这两个地址之一到多路开关选择器 58，当与当前取指地址关联的分支指令被假定为不被接受或“归于失败”而分

支指令到下一个顺序指令时，输出第一个地址。当与当前取指地址结合的分支指令被假设已被接受或指令流“跳到”一个新地址时，第二个地址被输出。

分支取指单元 52 有两个内部流水线，每个流水线计算这两个新取指地址之一。第一个分支取指单元流水线通过增加由 IFAR 62 锁存的地址计算顺序地址。在所描述实施例中这个第一分支取指单元流水线将 4 个指令长度加到 IFAR 62 的内容上去。第二分支取指单元流水线利用 IFAR 62 内容的一部分对被称为分支目标地址高速缓存 64（以下称 BTAC）的随机存取存储器（RAM）高速缓存的第一块进行变址。在描绘的实施例中，分支取指单元 52 存贮分支目标地址（取指地址）和与 BTAC64 中一些最近分支指令的取指地址相联系的分支指令的地址子集。

在描绘的实施例中，BTAC 64 是一个二路相关的高速缓存。BTAC 64 利用锁存在 IFAR 62 中的一个地址子集对 BTAC 64 中的两个入口进行变址。然后分支取指单元 60 将变址地址的剩余位同存贮在 BTAC 64 的两个变址项目中的每个里的地址位的子集进行比较。若两个比较中有一个匹配，那么出现一个 BTAC “命中”，且分支取指单元 52 结合匹配的地址位子集输出“取分支”取指地址。若两个比较均不匹配，那么产生一个 BTAC “未命中”，且分支取址单元 60 将已被增加的 IFAR62 的内容，输出到多路开关选择器 58。

分支调度/执行单元 54 当它接收到由 IFAR 62 的内容变址的特

定指令并且第一分支检测器 50 在四个取指指令中标识出第一分支指令时如果有的话立刻计算两个取指地址，分支调度/执行单元 54 在特定指令的调度阶段为特定分支指令计算两个地址。这两个取指地址相应于“不取分支”和“取分支”条件。分支调度/执行单元 54 仅将这两个地址之一输出到多路开关选择器 58。应当注意。分支取指单元 52 已经在流水线阶段根据同一分支指令将一新取指地址输出给多路开关选择器 58。根据在分支调度/执行单元 54 或分支完成单元 56 内同时被执行的分支指令的结果，IFAR62 可以用这两单元之一的输出装载。在这种情况下，数据处理器 12 将在这地址的开始取指令。

分支调度/执行单元 54 亦有两个内部流水线用以计算它的两个取指地址。第一流水线通过逐条增加分支指令地址来计算顺序地址。第一分支检测器 50 在每一包含 4 个被取指指令的组里计算第一分支指令地址，如果有的话。第二个分支调度/执行单元流水线根据特定分支指令计算取址地址。例如，这个流水线可以将一嵌入指令的偏移量加到分支指令地址中。一个分支预测单元（未示出）选择分支调度/执行单元 54 把两个地址中的一个输出到多路开关选择器 58。分支预测逻辑单元对应于用户可编程控制位，以下列三个方式之一进行操作：（1）动态分支预测；（2）用 BHT 更新的静态分支预测；（3）无 BHT 更新的静态分支预测。

一个分支指令是否被取这同分支转到何处是无关的（若被取的

话)。条件分支指令是分支指令的一类，它依靠一个在分支指令的调度/执行阶段通常是不知道的命令结果。这种依赖决定了分支是否被取，并不决定取指地址是什么，因此，分支调度/执行单元 54 根据动态分支预测方法或静态分支预测方法选择两个地址之一，这特定预测方法，通过在专用目的寄存器（未示出）设置某些位而使用户能控制。

根据动态分支预测方法进行操作的同时，分支调度/执行单元 54 利用 IFAR62 内容的一部分对被称为分支历史表 64（下文称 BHT）的第二 RAM 块进行变址，分支调度/执行单元 54 为每个分支指令或某些分支指令子集定义一个分支状态。特定分支指令的状态决定分支是否被取出。所描述的实施例中用的分支状态方式是已为大家熟知的含 4 种状态的模式：STRONG—NOT—TAKEN（强不取）WEAK—NOT—TAKEN（弱不取），WEAK—TAKEN（弱取）和 STRONG TAKEN（强取）。若指令分支状态对应于强不取或弱不取，分支调度/执行单元 54 预测分支将不取出。分支调度/执行单元 54，在指令完成后更新一特定分支指令的分支状态，若分支调度/执行单元 54 不正确地预测了分支指令，那么分支调度/执行单元 54 将更新 BHT 66 中相应的条目，使其从一个状态变为同样的弱状态或从弱状态到相反的弱状态。相反，若分支调度/执行单元 54 正确地预测了分支指令，那么，分支调度/执行单元 54 将更新 BHT 66 中相应条目，使其从一个弱状态变为相同强状态或从强状态变成同样的强状态。

根据静态分支预测方法进行操作的同时，分支调度/执行单元54用一位或多位嵌入到分支指令位中，以确定分支是否被实现。这些位在包含有分支指令的程序被编译完成时一次设定。所述实施例中，在静态分支预测模式中二位被用于预测分支指令。这两位之一是一个符号位，通常表示在分支调度/执行单元54中，第二流水线是加2个数以形成取指地址还是减2个数以形成取指地址（向前取指还是向后取址）。第二位是一个分支预测位，它表示分支是否应被实现。当符号位被触发时，分支预测位的逻辑状态和所得到的动作之间的关系颠倒，不管怎样，符号位和分支预测位被解码成单一实现/不实现位。若单一实现/不实现位在逻辑上等于第一状态，那么分支预测逻辑单元将选择在分支调度/执行单元54中由第一流水线产生的地址。若单一实现/不实现位逻辑上等于第二状态，那么分支预测逻辑单元将选择在分支调度/执行单元54中由第二流水线生成的地址。

当分支调度/执行单元54根据静态分支预测方法运行时，它可能会或可能不会更新BHT66。更新BHT66或不更新BHT 66的决定是一个有重要意义的决定。

分支完成单元56在特定分支指令被完成以后为一特定分支指令形成一个单一地址。同任何其它指令一样，当在分支指令之前的所有指令已把它的结果写到适当结构寄存器时，分支指令被完成。这时，分支指令所依据的条件，以及分支是否被实现已知道。须注意。

分支取指单元 52 已在同一分支指令的取指阶段根据同一分支指令将一指令取指地址输出给多路器 58。同时分支调度/执行单元 54 可能已在同一指令的调度/执行阶段将第二指令取指地址输出给多路开关选择器 58。

分支完成单元 56 在其内部有一先进先出队列，也就是分支队列 68，其中存贮着有关每个被执行的分支指令的数据。分支完成单元 56 通过数据/控制信号集 66 分支调度/执行单元 55 接收有关每个指令的某些数据：（1）由分支调度/执行单元 54 形成的但并不输出到多路开关选择器 58（未选择通路）的取指地址；（2）在调度/执行阶段进行分支计算所依赖的条件的预测值；（3）分支指令的分支状态。

分支完成单元 56 也存贮分支队列 68 中每个分支指令的地址。分支完成单元 56 从完成/调度单元 20 接收一个指示何时每个分支指令完成的控制信号。

在分支指令完成后，分支完成单元 56 接收分支条件的实在值，此分支条件是调度/执行取指地址的基础(这分支指令被分解了)。典型的这种条件是一个专用目的寄存器（条件寄存器）中可以被其它指令来修改的一位。若条件的实际值与存储在分支队列 68 中的预测值不同，则由分支调度/执行单元 54 形成的分支预测有错。在这种情况下，分支完成单元 56 与分支指令相结合输出所存贮的地址。这地址是一开始未被分支调度/执行单元 54 选择的路径。若条件的实

际值与存贮在分支队列68中预测值相同，那么由分支调度/执行单元54形成的分支预测是正确的。在这种情况下，分支完成单元56不需做任何事，除了与完成分支指令相结合作废在分支队列68中的条目以备后用。在任一种情况下，分支完成单元56根据预测与实际分支条件的比较和存贮的分支状态为分支指令产生一个新的分支状态，分支完成单元56以存贮的分支地址作为到BHT66的变址把新的分支状态传送到BHT66。分支完成单元56产生控制信号PENDING BRANCH（排起分支）该信号表示在分支队列68中如果有的话有多少未决定的分支指令。

地址选择器60决定大约4个输出地址中哪一个应使多路开关选择器58输出到IFAR 62。地址选择器60接收从分支取指单元52输出的取指地址，从分支调度/执行单元54输出的取指地址及从分支完成单元56来的控制信号，如果条件预测值和条件实际值不同分支完成单元56置位这个控制信号。若分支完成单元56置位其控制信号，那么地址选择器60会使多路开关选择器58输出由分支完成单元56形成的地址。若分支完成单元56没有置位其控制信号，且由分支调度/执行单元54生成的地址不同于在前面流水线阶段由分支取指单元52生成的地址，那么地址选择器60使多路开关选择器58输出由分支调度/执行单元54形成的地址。否则地址选择器60使多路开关选择器58输出由分支取指单元52形成的地址。若分支单元28发生故障；地址选择 器将输出IFAR 62的以前的值。应当明

白的是在任何时候，地址选择器60可以选择由4个不同分支指令形成的4个不同地址中一个。

猜测预取指控制单元70，从完成/调度单元30接收控制信号EXCEPTION（例外）及COMPLETION BUFFER EMPTY（完成缓冲器空），从调度缓冲器48接收DB EMPTY（DB空），从第一分支检测器50接收INCOMING BRANCH（引入分支）从分支完成单元56接收PENDING BRANCH（挂起分支），还接收地址选择器60的输出和三个预取指方式位。程序员所能存取的专用目的寄存器形成这三种方式位，猜测预取指控制单元70形成控制信号GO TO BUS（去总线）和OVERRIDE（取代）。

完成/调度单元30在形成例外指令的写回阶段置位EXCEPTION（例外）。当指令形成例外时数据处理器12立刻在存储器分支一地址，在存储器内常注例外操作子程序，在这些情况下，数据处理器12将不执行由IFAR 62变址的指令，这样，它需要预取指这些指令。否则，完成/调度单元30复位EXCEPTI ON（例外）。当完成缓冲器46为空时，完成/调度单元30置位COMPLETION BUFFER EMPT（完成缓冲空）否则，完成/调度单元30复位COMPLETION BVFFER EMPTY）。

第一或第0预取指方式位指示数据处理器12在实地址方式还是在虚地址方式。当在实地址方式时，数据处理器12直接用变址存储器生成地址，当在虚地址方式时，数据处理器12在变址存储器变

址前先转换它的地址。第 2 和第 3 预取指方式指示允许分支指令数目超过数据处理器 12 可预取指令的数目。若这两位为“00”，数据处理器 12 仅预取分支队列 68 中没有挂起分支指令；若为“01”，反预取分支队列 68 中不多于一个的挂起分支指令；若为“10”，仅预取分支队列 68 中不多于 2 个的挂起分支指令，若为“11”仅预取分支队列 68 中不多于 4 个的挂起分支指令。在这 4 种方式中，第一分支检测器 50 必须复位 INCOMING BRANCH（引入 分支）。

猜测预取指控制单元 70，当它适合对数据处理器 12 去预取指令时（如果请求指令不在指令高速缓存 24 中）。置位控制信号 GO TO BUS。当预指取方式位第 0 位设置到高逻辑态（实地址方式），调度缓冲器 40 置位 DB EMPTY，完成调度单元 30 置位 COMPLETION BUFFER EMPTY（完成缓冲器空）时，猜测预取指控制单元 70 置位 GO TO BUS。否则推测预取指控制单元 70 复位 GO TO BUS。在实地址寻址方式下，数据处理器 12 可能偶然地存取地址空间，这些空间一旦被存取将会永远改变（某些 I/0 设备）。因此，数据处理器 12 必须不在猜测存取这些地址空间。当预取指方式位第 0 位设置到低逻辑态时（虚地址方式），第一分支检测器 50 复位 INCOMING BRANCH 且在 PENDING BRANCH 小于或等于由第 2 和第 3 预取指方式位表示的值时，猜测预取指控制单元 70 置位 GO TO BUS。否则，猜测预取指控制单元 70 复位 GO TO BUS。

当分支单元 28 装载一新的取指地址到 IFAR 62 时，猜测预取指

控制单元 70 置位控制信号 OVER RIDE，这样优先使较早的流水线级输出。在这种情况下，前面取指地址不再需要。当地址选择器 60 从分支调度/执行单元 54 输出中或从分支完成单元 56 输出中选择一预取地址时，猜测预取指控制单元 70 置位 OVERRIDE。当地址选择器 60 从 IFAR 62 输出中或从分支取指单元 52 的输出中选择地址时，猜测预取指控制单元 70 复位 OVERRIDE。

图 4 显示了一个图 2 所示的指令高速缓冲存储器 24 的框图。指令高速缓存 24 具有一个含转换阵列 74 的存储器控制逻辑单元 72，一个高速缓存的随机存贮器（RAM）76，一个 ICACHE 控制逻辑单元 78 和一比较器 80。

存储器控制逻辑单元 72 从 IFAR62 接收虚拟地址的高有效位，转换阵列 74 在与未被转换的虚拟地址的低有效位结合的情况下，将所接收虚拟地址的高有效位转换，形成物理地址。存储器控制逻辑单元 72 将已被转换的高有效位作为一个控制信号 EXPECTED TAG。存储器控制逻辑单元 72 在实模式中不对所接收到的虚拟地址进行转换。在此模式中，存储器控制逻辑单元 72 仅将所接收的地址传送给比较器 80 和 ICACHE 控制逻辑 78。

高速缓存 76 存贮大量频繁使用指令的数据和相关标签（tags）对。高速缓存 76 接收由 IFAR62 产生的虚拟地址的低有效位，并对一对指令数据和标签建立索引。在逻辑上被索引的标签与所存贮指令数据原始地址的物理地址的高有效位相等。

比较器 80 从高速缓冲存贮器 72 接收被索引的存贮标签，从存储器控制逻辑单元 72 接收 EXPECTED TAG。如果这两个多位信号是相同的，则被 IFAR 62 的内容变址的指令存在于指令高速缓存 24 中。数据处理器 12 不必从主存贮块 16 中取指令，在这种情况下，比较器 80 置位一个控制信号 HIT /MISS（命中/未命中）。如果这两个多位信号不同，那么由 IFAR 62 内容变化的指令不在指令缓存器 24。在此情况下，比较器 80 复位一个控制信号 HIT MISS（命中/未命中）。同时，数据处理器 12 将必须从主存贮块 16 中取出这个指令。本发明所涉及的就是这一取操作的时序。

ICACHE 控制逻辑单元 78 控制着指令高速缓存 16 与 BIU23 的界面。更具体地说，ICACHE 控制逻辑单元 78 将指令请求送至 BIU 23，从 BIU 23 接收所请求的指令。并把接收到的指令存贮于高速缓存 76（路径未示出）。ICACHE 控制逻辑单元 78 产生控制信号 ADDRESS（地址），LOAD（装载），REQUEST（请求）和 COMMIT，并从 BIU 23 接收控制信号 DATA（数据），DATA VALID（数据有效），DATA COMPLETE（数据完成）和 CANCEL ENABLE（取消使能），ICACHE 控制逻辑单元 78 还从分支单元 28 接收 GO TO BUS（去总载）和 OVERRIDE（取代）信号，从存储器控制逻辑单元 72 接收 EXPECTED TAG，比较器 80 接收 HIT MISS 信号，从 IFAR 62 接收所请求指令的虚拟地址。

ICACHE 控制逻辑单元 78 将 EXPECTED TAG 与 IFAR 62 的内

容的低有效位连接以产生 ADDRESS，一个不存贮于指令高速缓存 24 的指令的地址。当其有一有效的指令要求时，ICACHE 控制逻辑单元 78 置位 LOAD REQUEST（装载请求）。当适合将请求传送到主存贮块 16 时，ICACHE 控制逻辑单元置位 COMMIT。下面将结合图 5 对 LOAD REQUEST（装载请求）和 CMMIT 进行说明。DATA 包括一个来自主存贮块 16 的所请求的指令。DATA VALID（数据有效）指示 DATA 所含的数据是合法的。BIU 在其将最后一节指指令数据传送至指令高速缓存 24 时置位 DATA COMPLETE（数据完成）CANCEL ENABLE（取消使能）指示 ICACHE 控制逻辑单元 78 可以取消一个请求。

CANCEL ENABLE 在实施例中很有用，就如图 1 所示，在此数据处理器 14 在其向主存贮器单元 16 请求数据之前先从外部高速缓存器 22 请求数据。在此实施例中，外部高速缓存 22 和主存贮器单元 16 是通过独立的或成对的总线被存取的。在此实施例中，数据处理器 14 仅在所请求的指令不在外部高速缓存 22 中时垄断公共地址和数据总线。如上所述。分支单元 28 在某些情况下确定其不需要所请求的指令。在这些情况下，如果数据处理器 14 已经对所请求的指令寻找了外部高速缓存 22 但是未存取主存贮单元 16，ICACHE 控制逻辑单元 78 将复位 LOAD REQUEST（装载请求）。

图 5 显示了图 4 所示的指令高速缓存逻辑单元 78 的状态转换图。ICACHE 控制逻辑单元 78 保持在 IDLE（闲置）状态，直到分支

单元 28 请求一个不在指令高速缓存 24 中的指令，即一个“高速缓存未命中”。在高速缓存未命中之后，指令高速缓存逻辑单元 78 将请求一个新指令。如果比较器 80 复位 HIT /MISS（命中/未命中）且猜测预取指令控制单元 70 复位 GO TO BUS（去总线），则 ICACHE 控制逻辑单元 78 转换到 MISS OUTSTANDING（明显未命中）状态、最终，指令高速缓存逻辑单元 78 将或者提交这个请求，如果预测预取指令控制单元 70 置位 GO TO BUS（去总线），或者取消该请求，如果预测预取指令控制单元 70 置位 OVERRIDE（取代）。应当明白的是，在某些场合，即使在预测预取指控制单元 70 置位 OVERRIDE（取代）之后，主存贮单元 16 仍将把指令返回给数据处理器 12。在这些场合下，ICACHE 控制逻辑单元 78 将忽略来自主存贮单元 16 的指令数据。

在那些指令高速缓存逻辑单元 78 确认请求的场合下，当预测预取指控制单元 70 置位 GO TO BUS（去总线）时，ICACHE 控制逻辑单元 78 从 MISS OUTSTANDING（明显未命中）状态转换为 COMMITTED 运行状态。如果比较器 80 复位 HIT /MISS（命中/未命中）且预测预取指控制单元 70 置位 GO TO BUS（去总线）ICACHE 控制逻辑单元 78 可从 IDLE（闲置）状态直接转换为 COMMTTED 运行状态。如果 BIU 23 置位 DATA COMPLETE（数据完成），ICACHE 控制逻辑单元 78 将从 COMMITTED（运行）状态变回 IDLE（闲置）状态。

在那些指令高速缓存逻辑单元 78 取消请求的场合，如果预测预

取指控制单元 70 置位 OVERRIDE（取代）且 BIU 23 置位 CANCEL ENABLE（取消使能），ICACHE 控制逻辑单元 78 将从 MISS OUTSTANDING（明显未击中）状态直接转换为 IDLE（闲置）状态。当预测预取指控制单元 70 置位 OVERRIDE（取代）且 BIU 23 复位 CANCEL ENABLE（取消使能）时，ICACHE 控制逻辑单元 78 将从 MISS OUTSTANDING（明显未击中）状态转换成 OVER—RIDDEN（已被取代）状态。如果 BIU 23 置位 CANCEL ENABLE（取消使能）或 DATA COMPLETE（数据完成），ICACHE 控制逻辑单元 78 将从 OVER—RIDDEN（已被取代）状态变回 IDLE（闲置）状态。

控制信号 COMMIT 和 LOAD REQUEST（装载请求）的逻辑状态由图 5 所示的状态转换图决定。更具体而言，ICACHE 控制逻辑单元 78 在其处于 COMMITTED（运行）状态时置位 COMMIT（运行信号）。否则，ICACHE 控制逻辑单元 78 复位 COMMIT（运行信号）、当处于 MISS OUTSTANDING（明显未击中）、COMMITTED（运行）或 OVER—RIDDEN（已被取代）当中任一状态时，ICACHE 控制逻辑单元 78 置位 LOAD REQUEST（装载请求）。当处于闲置状态时，ICACHE 控制逻辑单元 78 复位 LOAD REQUEST（装载请求）。

图 6 显示了图 2 所示的 BIU 23 的方块图。与本发明相关的 BIU 23 的部分包括一个 BIU 仲裁器 82，一个 BIU 控制逻辑单元 84 和一个多路器（标为 MUX）86。

BIU 仲裁器 82 从指令高速缓存 24、数据高速缓存 26 和由指令

与数据高速缓存器共享的 tablewalk 电路（未示出）接收存取外部地址和数据总线的请求 tablewalk 电路将数据装载入用于地址转换的转换阵列 74。BIU 仲裁器 82 根据一个与本发明无关的协议决定哪个请求将在何时实际存取地址和数据总线。BIU 仲裁器产生一个控制信号 SELECT（选择），用于选择哪一个多路器 86 的输入被送往地址总线。一独立数据路径（未示出）把与数据高速缓冲装载或存贮操作相关的数据送到外部数据总线。多路器 86 从三个可能请求 者中的每个接收一个地址。

BIU 控制逻辑 84 如同 BIU 仲裁器 82 所做的，接收相同的三个存取外部地址和数据总线的请求。BIU 控制逻辑 84 还从地址和数据总线接收控制信号并产生与地址和数据总线相关的控制信号。举例说，BIU 控制逻辑 84 产生一个输出给外部总线的数据有效信号 BUS VALID（总线有效），并且从外部总线接收一个类似的信号。当三个可能请求者中的至少一个提出请求时，BIU 控制逻辑单元 84 置位 BUS VALID（总线有效）。同时，如果被选择的请求产生于指令高速缓存 24，那么必须在 BIU 控制逻辑单元 84 置位 BUS VALID（总线有效）之前同时置位 LOAD REQUEST（装载请求）和 COMMIT（运行信号）。BIU 控制逻辑单元 84 把从外部总线接收到的数据有效信号作为 DATA VALID（数据有效）传送给指令高速缓存 24。在本发明的第一个实施例中，BIU 控制逻辑单元 84 总是置位 CANCEL ENABLE（取消使能）。最后，当其从外部总线接收到最后一节指数据时，

BIU 控制逻辑单元 84 置位 DATA COMPLETE（数据完成）。

如上所述，本发明可以插入这样一种具有一个需要通过不同于寻常的总线对才可存取的数据处理器和外部高速缓存的数据处理系统。在这种情况下，上述的某些信号和功能稍有不同。更具体地说，BIU 控制逻辑单元 84 首先将 IFAR 62 的内容传送至外部高速缓存块 22 而不理会分支队列 68 中未决定分支指令的个数。应当注意，指令高速缓冲存贮器 24 已经对 IFAR62 变化的指令进行了寻找。这一对独立总线的总线 activity 不会显著影响系统效能。在这一步中，BIU 控制逻辑单元 84 置位一个控制信号 SECONDARY CACHE VALID（二级高速缓存有效）。如果指令不在外部高速缓存存贮块 22（一个“L2”未命中），那么 BIU 控制逻辑单元 84 将通过上述的公共总线存取主存贮块 16。BIU 控制逻辑单元 84 在数据处理器存取外部高速缓存之后置位 CANCEL ENABLE（取消使能）。

尽管本发明已参照特殊实施例而被说明，那些熟悉本领域技术的人仍可做进一步的改进和提高。例如，本发明可被插入那些习惯上被归为复杂指令集计算机或 CISC 机器的数据处理器。此外，某些功能单元可在某些实施例中被省略或是重新置于数据处理 12 的其它区域。所以应当明白，本发明包括了所有这些未偏离其精髓和附加权利要求所限定的发明的范围的改进。

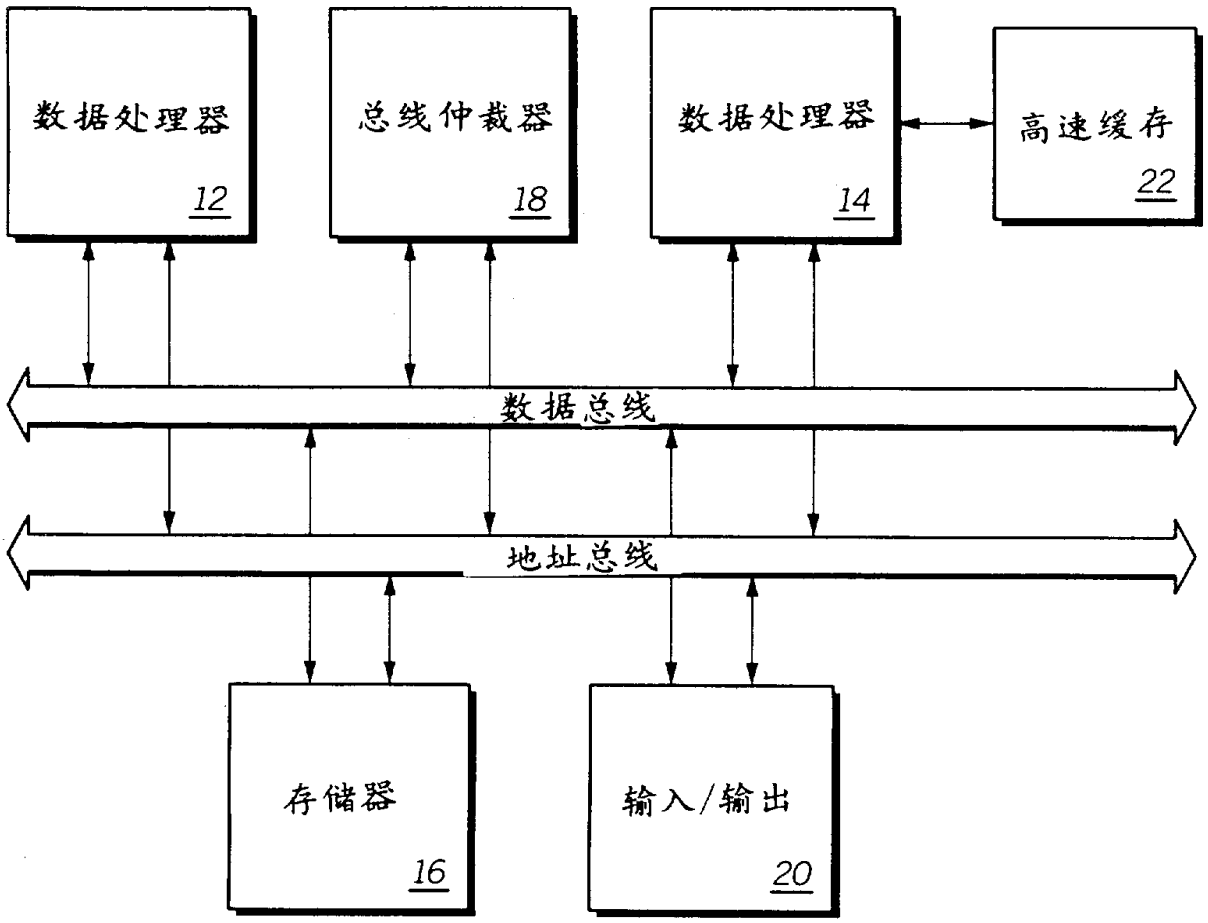
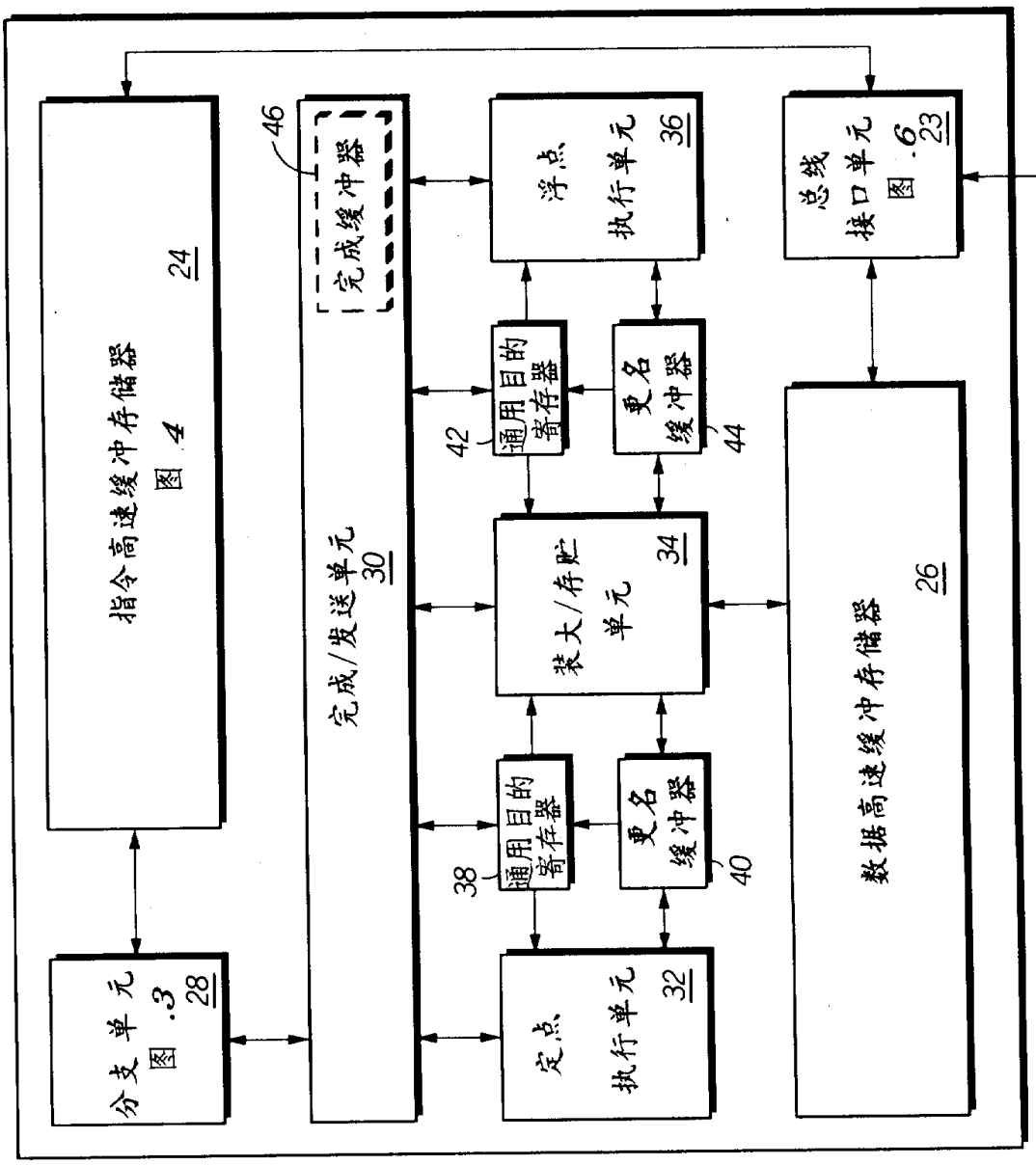


图1


指令高速缓冲存储器
图 4
24
分支单元
图 3
28
完成/发送单元
30
46
完成缓冲器
浮点
执行单元
36
总线
接口单元
图 6
23
通用目的
寄存器
42
更名
缓冲器
44
装入/存贮
单元
34
通用目的
寄存器
38
更名
缓冲器
40
定点
执行单元
32
数据高速缓冲存储器
26
12
图 2
121

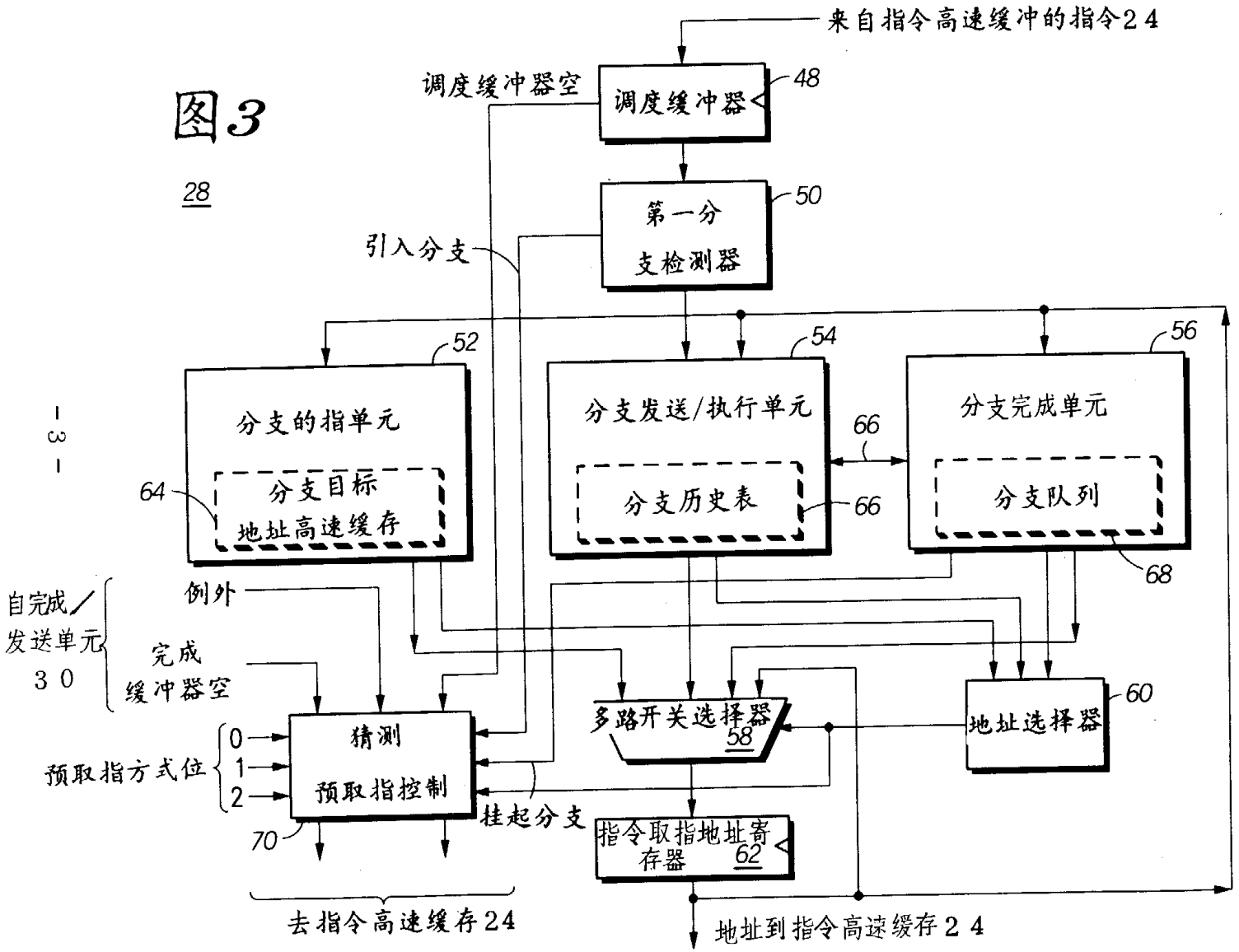

来自指令高速缓冲的指令24
调度缓冲器空
调度缓冲器
48
第一分支检测器
50
引入分支
28
52
分支的指单元
分支目标地址高速缓存
64
54
分支发送/执行单元
分支历史表
66
66
56
分支完成单元
分支队列
68
自完成/发送单元30
例外
完成缓冲器空
猜测预取指控制
预取指方式位
0
1
2
70
多路开关选择器
58
地址选择器
60
挂起分支
指令取指地址寄存器
62
去指令高速缓存24
地址到指令高速缓存24

图3

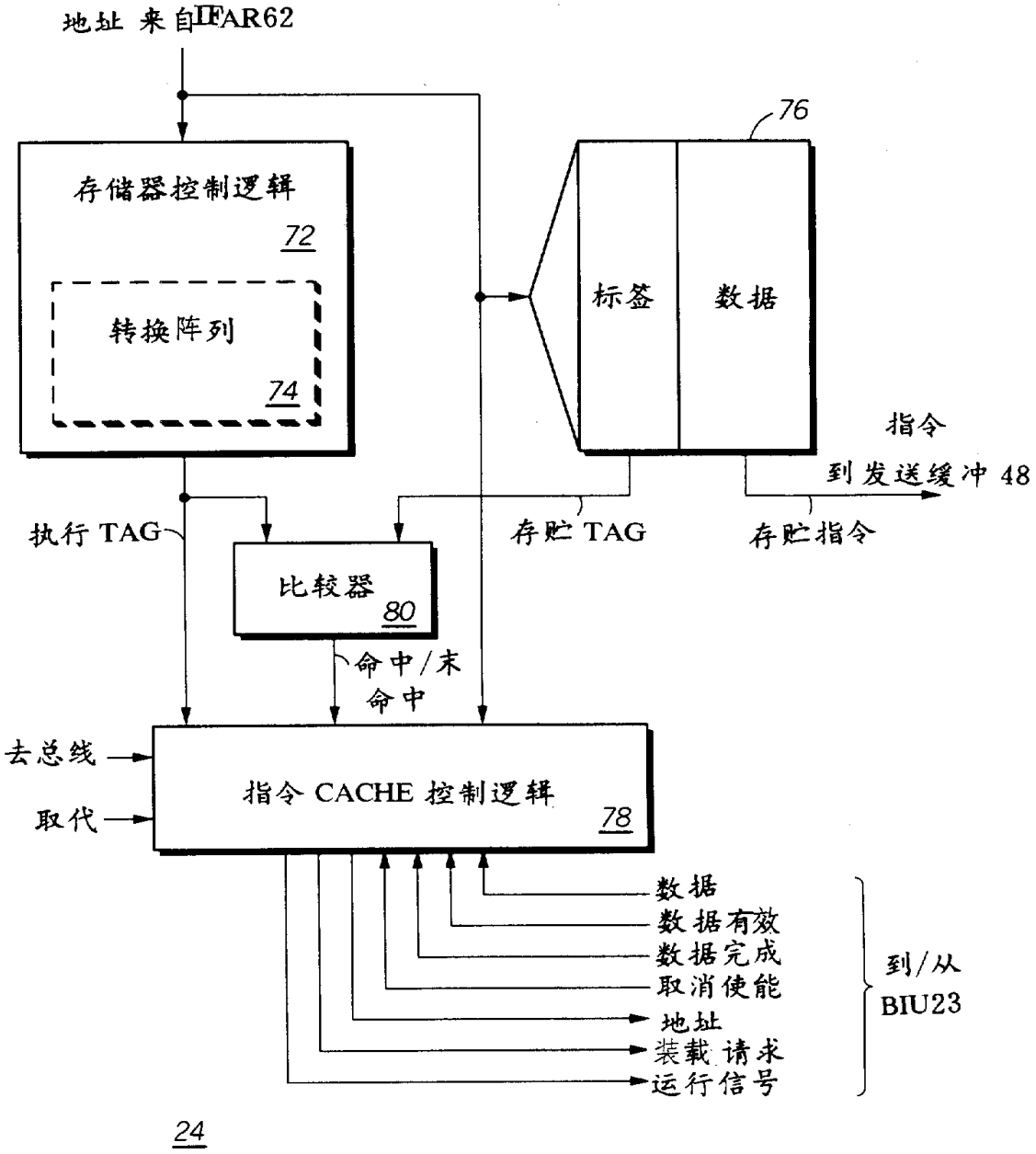

地址 来自IFAR62
存储器控制逻辑
72
转换阵列
74
76
标签
数据
指令
到发送缓冲 48
执行 TAG
存贮 TAG
存贮指令
比较器
80
命中/未命中
去总线
取代
指令 CACHE 控制逻辑
78
数据
数据有效
数据完成
取消使能
地址
装载请求
运行信号
到/从 BIU23
24

图4

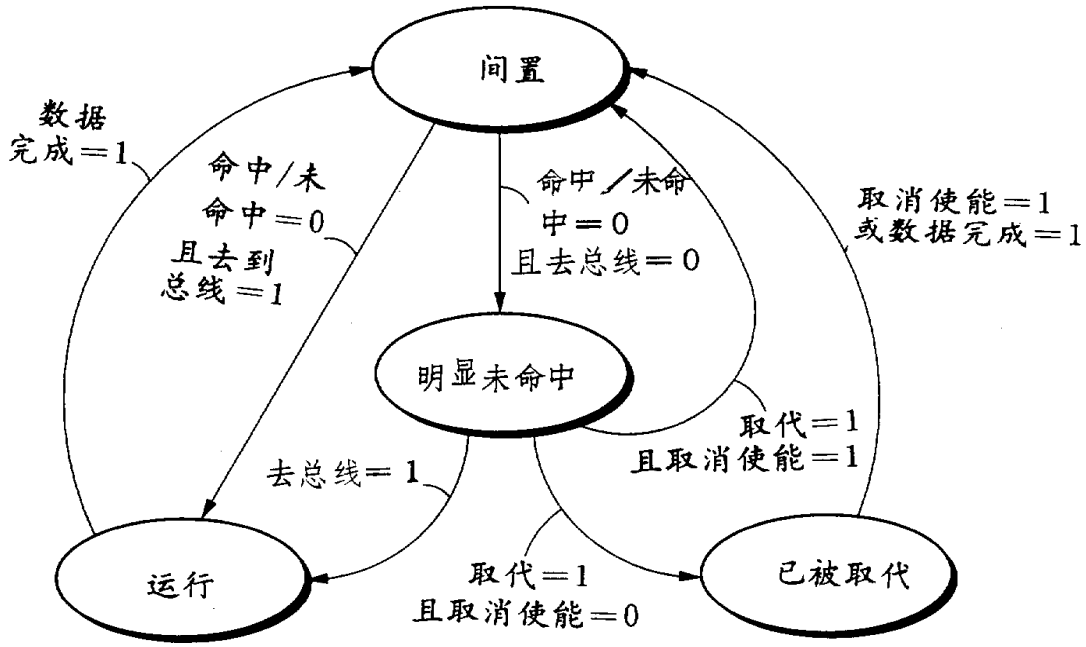

间置
数据
完成＝1
命中／未
命中＝0
且去到
总线＝1
命中／未命
中＝0
且去总线＝0
取消使能＝1
或数据完成＝1
明显未命中
取代＝1
且取消使能＝1
去总线＝1
运行
取代＝1
且取消使能＝0
已被取代

图5

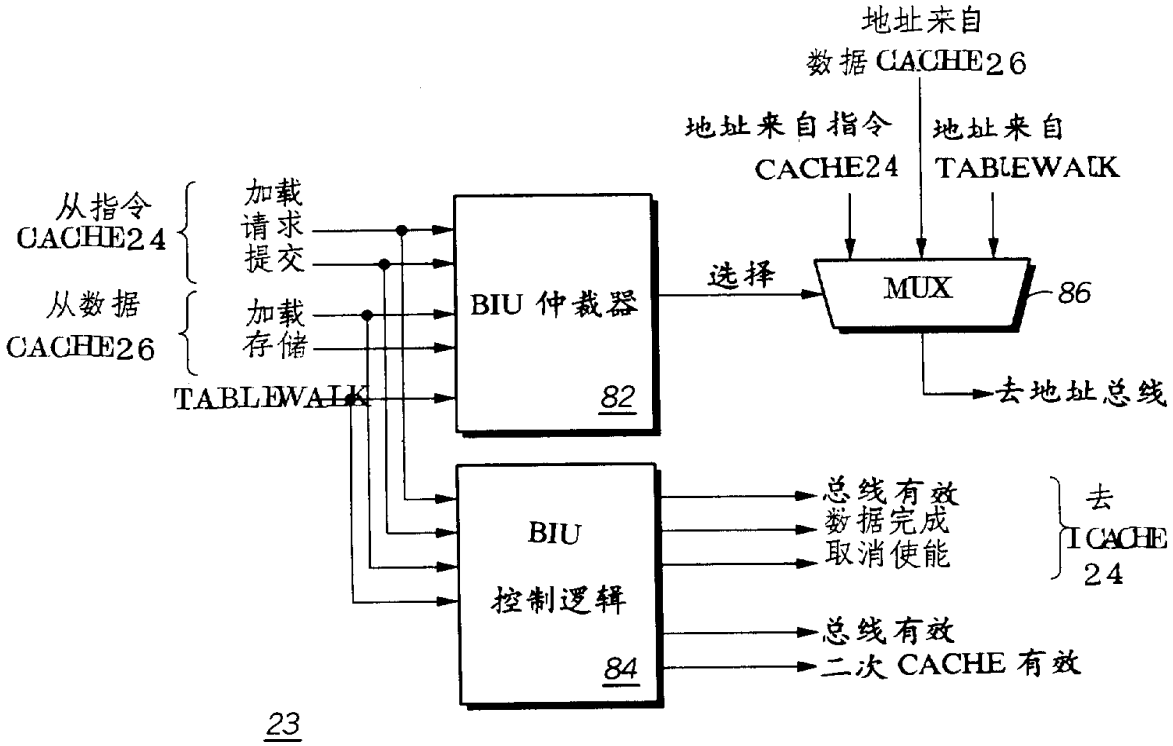

地址来自
数据 CACHE26
地址来自指令
CACHE24
地址来自
TABLEWALK
从指令
CACHE24
加载
请求
提交
从数据
CACHE26
加载
存储
TABLEWALK
BIU 仲裁器
82
选择
MUX
86
去地址总线
BIU
控制逻辑
84
总线有效
数据完成
取消使能
去
ICACHE
24
总线有效
二次 CACHE 有效
23

图6