(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 103336681 A
(43)申请公布日 2013.10.02

(21)申请号 201310277005.4

(22)申请日 2013.07.03

(71)申请人 同济大学
地址 200092 上海市杨浦区四平路 1239 号

(72)发明人 吴俊 骆原 苏立峰 陈伟
沈嘉琦 李思昌 周文宗

(74)专利代理机构 上海科盛知识产权代理有限公司 31225
代理人 宣慧兰

(51)Int.Cl.
G06F 9/30(2006.01)
G06F 9/38(2006.01)

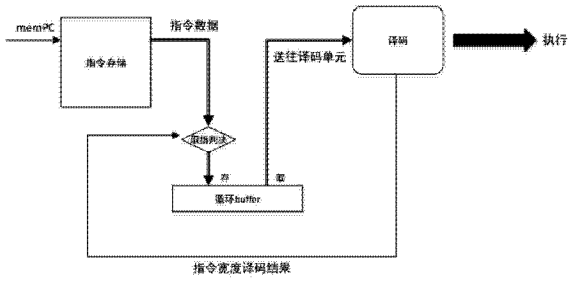
权利要求书1页 说明书3页 附图3页

(54)发明名称

针对采用变长指令集的流水线结构处理器的取指方法

(57)摘要

本发明涉及一种针对采用变长指令集的流水线结构处理器的取指方法，采用不同宽度的指令在指令存储器中连续存储，分别设置取指指针和译码指针，用来保存下一条需要取指的地址和下一条需要译码的地址。取指部件中设置一定宽度的指令数据循环缓冲区，指令存储器中取出来的数据需覆盖缓冲区的特定部分，取指阶段从缓冲区中的特定位置取出数据进行译码。最后根据译码结果以及两个指针决定下一个周期是否需要进行取指。与现有技术相比，本发明可以解决变长指令集的取指问题，同时保证指令存储器深度不增加以及处理器流水线的吞吐量不降低。



CN 103336681 A

1. 一种针对采用变长指令集的流水线结构处理器的取指方法，其特征在于，包括：

流水线的取指级，包含指令存储器、取指地址寄存器、取指判决单元；

流水线的译码级，包含指令译码模块、译码地址寄存器位；

指令数据循环缓冲区，该指令数据循环缓冲区为流水线的取指级和流水线的译码级之间的流水线寄存器；

其中，所述的指令存储器的宽度为单条指令最大长度，且不同长度的指令在指令存储器中连续存储，所述的指令译码模块对指令数据进行译码，并通过对取指地址寄存器和译码地址寄存器进行逻辑判断，决定下一个周期是否取指，将结果反馈至流水线的取指级，所述的指令数据循环缓冲区用于缓存取入的指令数据，并通过取指地址寄存器和译码地址寄存器控制指令数据的更新。

2. 根据权利要求 1 所述的一种针对采用变长指令集的流水线结构处理器的取指方法，其特征在于，所述的指令存储器满足以下条件：

a、指令的长度满足表达式：2^x 字，$x > 0$，字长任意，其中最长的指令长度为 M 个字，$M = 2^m$，最短的指令长度为 N 个字，$N = 2^n$，每条指令的长度介于 N，M 之间，不同长度的单条指令所需要执行周期数相同；

b、不同长度的指令在指令存储器中连续存储；

c、取指地址寄存器保存下一次进行取指的地址，每次取指后其需自增 M。

3. 根据权利要求 1 所述的一种针对采用变长指令集的流水线结构处理器的取指方法，其特征在于，所述的指令译码模块满足以下条件：

a、译码地址寄存器保存下一个需要进行译码的指令的起始地址，每次译码结束后其需自增该指令的真实长度，

b、指令译码模块从指令数据循环缓冲区中取出指令数据进行译码后送入执行模块，流水线的译码级根据自增后的译码地址寄存器以及流水线的取指级的取指地址寄存器的比较结果，决定下一个周期是否取指，并将决定结果反馈至取指级；

c、取指判决单元接收到译码级反馈的比较结果，决定是否向指令内存请求数据，以及是否向指令数据循环缓冲区中写入新的数据。

4. 根据权利要求 1 所述的一种针对采用变长指令集的流水线结构处理器的取指方法，其特征在于，所述的指令数据循环缓冲区满足以下条件：

a、指令数据循环缓冲区的宽度为设置为 2M，取指地址寄存器决定每次取入的指令数据所覆盖的指令数据循环缓冲区中的位置，译码地址寄存器决定每次需要译码的指令数据在指令数据循环缓冲区中的位置，取指地址寄存器和译码地址寄存器之间的相对位置决定指令数据循环缓冲区中有效的数据量；

b、译码级取指判断逻辑为：当 memPC == PC 时，指令缓冲区为空或者均为无效数据，则下个周期可以取指；当 memPC ! = PC，且 memPC-PC ∈ [2*M, M+N) 时，则指令缓冲区中没有足够的空间来缓存 M 个字，故无法取指；当 memPC ! = PC，且 memPC-PC ∈ [M+N,0) 时，下一个周期可以进行取指；其中 memPC 为取指地址寄存器，PC 为译码地址寄存器，M 为最长的指令长度，N 为最短的指令长度。

针对采用变长指令集的流水线结构处理器的取指方法

技术领域

[0001] 本发明涉及一种取指方法，尤其是涉及一种针对采用变长指令集的流水线结构处理器的取指方法。

背景技术

[0002] 所谓指令集，就是 CPU 中用来计算和控制计算机系统的一套指令的集合，而每一种新型的 CPU 在设计时就规定了一系列与其他硬件电路相配合的指令系统。而指令集的先进与否，也关系到 CPU 的性能发挥，它也是 CPU 性能体现的一个重要标志。

[0003] 在技术飞快发展的现代，由于 ASIP 的设计越来越复杂，指令集的设计也不再局限于传统的 RISC 模式，VLIW 和 SIMD 等技术也已经广泛应用于指令集设计当中。VLIW :(Very Long Instruction Word，超长指令字）一种非常长的指令组合，它把许多条指令连在一起，增加了运算的速度。SIMD :(Single Instruction Multiple Data，单指令多数据流）能够复制多个操作数，并把它们打包在大型寄存器的一组指令集，以同步方式，在同一时间内执行同一条指令。在通用处理器设计的过程中，指令集往往是定长的，而在针对特殊领域的处理器的设计中，指令集的设计也渐渐复杂，为了节省指令存储器的空间，降低其压力，最好的方法是设计变长指令集，对于复杂功能的指令，如 VLIW、SIMD 等类型的指令，可分配较长的指令长度，对于简单功能的指令可以分配较短的指令长度。这时除了处理器的处理单元，译码单元需要重新设计，处理器的取指策略也需要重新考量，在不增加指令存储器深度以及不降低处理器流水线的吞吐量的条件下，保证取指的正确。

发明内容

[0004] 本发明的目的就是为了克服上述现有技术存在的缺陷而提供一种针对采用变长指令集的流水线结构处理器的取指方法，该方法可以解决变长指令集的取指问题，同时保证指令存储器深度不增加以及处理器流水线的吞吐量不降低。

[0005] 本发明的目的可以通过以下技术方案来实现 ：

[0006] 一种针对采用变长指令集的流水线结构处理器的取指方法，包括 ：

[0007] 流水线的取指级，包含指令存储器、取指地址寄存器、取指判决单元 ；

[0008] 流水线的译码级，包含指令译码模块、译码地址寄存器位 ；

[0009] 指令数据循环缓冲区，该指令数据循环缓冲区为流水线的取指级和流水线的译码级之间的流水线寄存器 ；

[0010] 其中，所述的指令存储器的宽度为单条指令最大长度，且不同长度的指令在指令存储器中连续存储，所述的指令译码模块对指令数据进行译码，并通过对取指地址寄存器和译码地址寄存器进行逻辑判断，决定下一个周期是否取指，将结果反馈至流水线的取指级，所述的指令数据循环缓冲区用于缓存取入的指令数据，并通过取指地址寄存器和译码地址寄存器控制指令数据的更新。

[0011] 所述的指令存储器满足以下条件 ：

[0012] a、指令的长度满足表达式：2^x 字，$x > 0$，字长任意，其中最长的指令长度为 M 个字，$M = 2^m$，最短的指令长度为 N 个字，$N = 2^n$，每条指令的长度介于 N，M 之间，不同长度的单条指令所需要执行周期数相同；

[0013] b、不同长度的指令在指令存储器中连续存储；

[0014] c、取指地址寄存器保存下一次进行取指的地址，每次取指后其需自增 M。

[0015] 所述的指令译码模块满足以下条件：

[0016] a、译码地址寄存器保存下一个需要进行译码的指令的起始地址，每次译码结束后其需自增该指令的真实长度，

[0017] b、指令译码模块从指令数据循环缓冲区中取出指令数据进行译码后送入执行模块，流水线的译码级根据自增后的译码地址寄存器以及流水线的取指级的取指地址寄存器的比较结果，决定下一个周期是否取指，并将决定结果反馈至取指级；

[0018] c、取指判决单元接收到译码级反馈的比较结果，决定是否向指令内存请求数据，以及是否向指令数据循环缓冲区中写入新的数据。

[0019] 所述的指令数据循环缓冲区满足以下条件：

[0020] a、指令数据循环缓冲区的宽度为设置为 2M，取指地址寄存器决定每次取入的指令数据所覆盖的指令数据循环缓冲区中的位置，译码地址寄存器决定每次需要译码的指令数据在指令数据循环缓冲区中的位置，取指地址寄存器和译码地址寄存器之间的相对位置决定指令数据循环缓冲区中有效的数据量；

[0021] b、译码级取指判断逻辑为：当 memPC == PC 时，指令缓冲区为空或者均为无效数据，则下个周期可以取指；当 memPC ! = PC，且 memPC-PC ∈ [2*M，M+N) 时，则指令缓冲区中没有足够的空间来缓存 M 个字，故无法取指；当 memPC ! = PC，且 memPC-PC ∈ [M+N，0) 时，下一个周期可以进行取指；其中 memPC 为取指地址寄存器，PC 为详码地址寄存器，M 为最长的指令长度，N 为最短的指令长度。

[0022] 与现有技术相比，本发明。

附图说明

[0023] 图 1 为本发明整体的取指结构图；

[0024] 图 2 为从指令存储中取指并存入指令数据循环缓冲区的流程图；

[0025] 图 3 为从循环缓冲区中取数据进行译码，并判决下一个周期是否取指的流程图。

具体实施方式

[0026] 下面结合附图和具体实施例对本发明进行详细说明。

[0027] 实施例

[0028] 如图 1 ～ 3 所示，一种针对采用变长指令集的流水线结构处理器的取指方法，包括流水线的取指级、译码级以及指令数据循环缓冲区 (Instruction Buffer，IB)。其中，取指级包含：指令存储器 (Instruction Memory，IM)、取指地址寄存器 (memPC)、取指判决单元 (Fetch Decision，FD)；流水线译码级包含：指令译码模块 (Instruction Decoder，ID)、译码地址寄存器位 (PC)；指令数据循环缓冲区为取指级和译码级之间的流水线寄存器的一部分。

[0029] 其中，指令存储器的宽度为单条指令最大长度，且不同长度的指令在指令存储器中连续存储。以指令长度为 4 字节和 8 字节两种为例，指令存储器的字长为 8 字节。即每次取 64 位的指令数据。而由于，每次“消耗”的指令内容，可能为 4 字节或 8 字节两种情况。故需要一个指令缓冲 IB 来进行缓存指令数据，并需要逻辑控制本周期是否要取指，否则缓冲 IB 就会溢出。IB 设置为 16 字节的缓冲区。故可知需要两个 PC 来联合控制。一个是 memPC 它准确地描述下一个需要取进 IB 的 8 字节为指令数据的地址。一个是 PC，它用来准确地指定下一个需要执行的指令的地址。可见通过 memPC 和 PC 相对位置的比较，可以推算出当前 IB 中的可用的指令数据剩余数量，以决定当前 IB 中是否有足够空间存放下一条 8 字节的指令数据。这种比较判断明显是在译码阶段做出，因为需要用来参与判断的 PC 应该是译码后才能知道自增的步长为 4 还是 8。

[0030] memPC 和 PC 的判断要分成两种情况：

[0031] 1、若 memPC ＝＝ PC，这时应该是刚开机，或者刚跳转，或刚被重置后的情况，这时需要进行取指。

[0032] 2、若 memPC ！ ＝ PC 且 memPC-PC ＝＝ 16 字节，这时 IB 中的数据都是取进来且未被使用过的数据。这时取指需要暂停，否则 IB 会溢出。

[0033] 3、其他的情况都是需要进行取指的。这里有一个特殊的情况，就是 memPC 和 PC 相差 12 字节，这时只有 4 字节的缓冲空可用，但是由于指令长度至少位 4 字节，接下来至少会有 4 字节的数据被消费，故还是可以取指的。

[0034] memPC 和 PC 的比较结果从译码阶段反馈至取指阶段。

[0035] 本发明所主张的权利范围并不局限于此。本发明还有其他多种实施例，在不背离本发明精神及其实质的情况下，本领域技术人员可根据本发明作出各种相应的改变和变形，但这些改变和变形都应属于本发明所附的权利要求的保护范围。

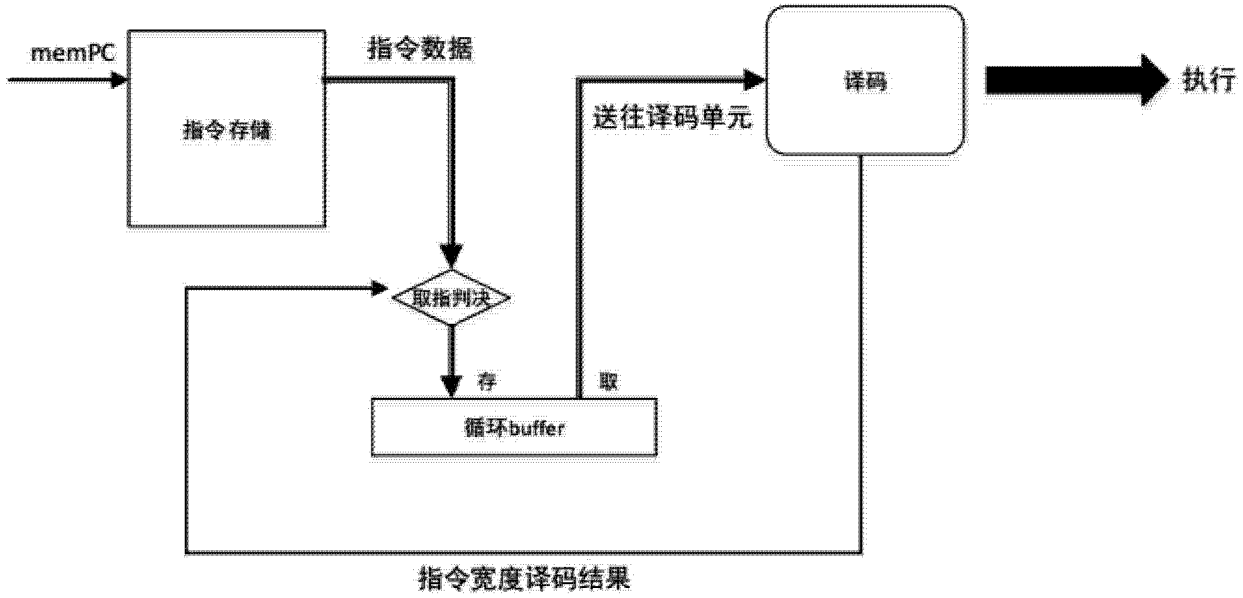


图 1

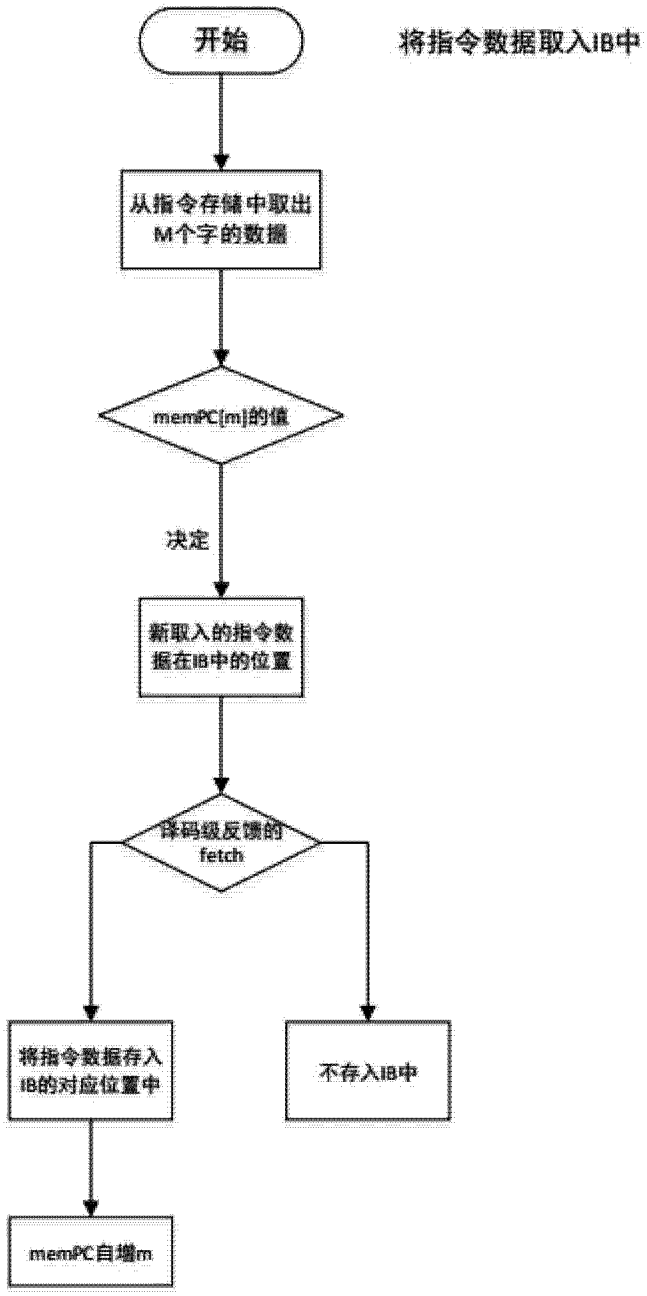

开始
将指令数据取入IB中
从指令存储中取出M个字的数据
memPC[m]的值
决定
新取入的指令数据在IB中的位置
译码级反馈的fetch
将指令数据存入IB的对应位置中
不存入IB中
memPC自增m

图 2

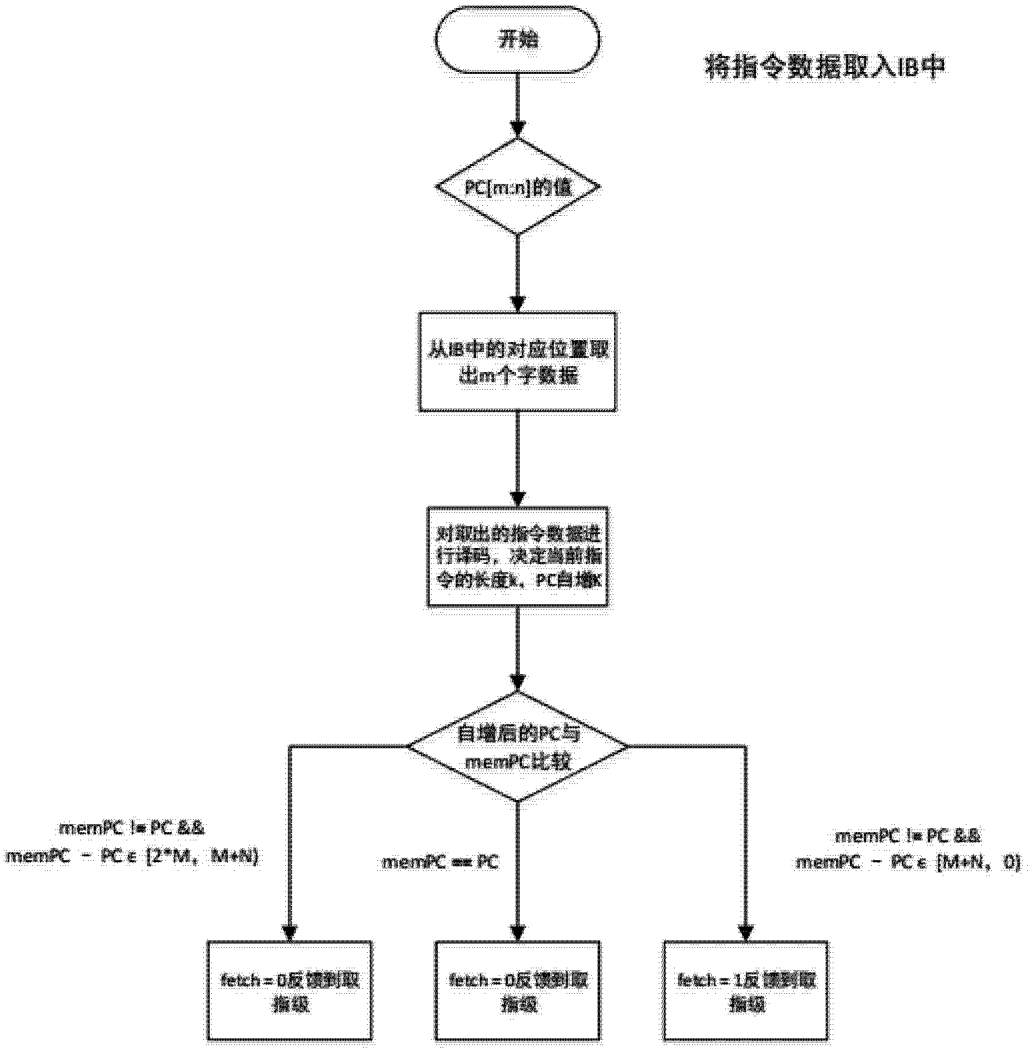

开始
将指令数据取入IB中
PC[m:n]的值
从IB中的对应位置取出m个字数据
对取出的指令数据进行译码，决定当前指令的长度k，PC自增K
自增后的PC与memPC比较
memPC != PC &&
memPC - PC ∈ [2*M，M+N)
memPC == PC
memPC != PC &&
memPC - PC ∈ [M+N，0)
fetch = 0反馈到取指级
fetch = 0反馈到取指级
fetch = 1反馈到取指级

图 3