(19)国家知识产权局

(12)发明专利

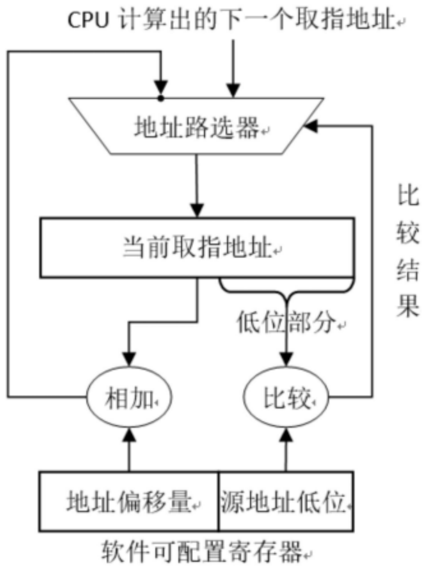
(10)授权公告号 CN 111124494 B
(45)授权公告日 2023.07.25

(21)申请号 201911301722.X
(22)申请日 2019.12.17
(65)同一申请的已公布的文献号
申请公布号 CN 111124494 A
(43)申请公布日 2020.05.08
(73)专利权人 天津国芯科技有限公司
地址 300457 天津市滨海新区开发区第四大街80号天大科技园软件大厦北楼306室
(72)发明人 王粟　肖佐楠　郑茳
(74)专利代理机构 天津滨海科纬知识产权代理有限公司 12211
专利代理师 耿树志
(51)Int.Cl.
G06F 9/30(2006.01)
G06F 9/448(2018.01)
(56)对比文件
CN 104407968 A,2015.03.11
CN 109582364 A,2019.04.05
CN 101571818 A,2009.11.04
CN 104424129 A,2015.03.18
CN 107943727 A,2018.04.20
US 2006/0095744 A1,2006.05.04
CN 103019655 A,2013.04.03
审查员 姚希

权利要求书1页　说明书4页　附图2页

(54)发明名称
一种CPU中加速无条件跳转的方法及电路

(57)摘要
本发明提供了一种使用在嵌入式CPU中的，用于加速程序中无条件跳转的方法及其电路实现。该方法通过在CPU中，加入软件可寻址寄存器及相应的取指计算与控制电路，从而实现通过寄存器访问指令，直接无缝切换CPU取指流水线，达到在绝大多数情况下加速无条件跳转的效果。



1.一种CPU中加速无条件跳转的方法,其特征在于,包括如下步骤:

1)首先存储源地址低位值和目的地址偏移量;

2)在CPU运行的每个时钟周期,将存储的源地址低位值与CPU的当前取指地址的等位宽的低位部分进行比较得到比较结果;

3)在CPU运行的每个时钟周期,将目的地址偏移量与CPU的当前取指地址进行加法运算得到相加结果;

4)在CPU运行的每个时钟周期,判断比较结果是否为相等:

若相等,则将步骤3)得到的相加结果作为最终地址进行输出,并在下个时钟周期,将当前取指地址更新为相加结果;

若不等,则将CPU计算出的下一个取指地址作为最终地址进行输出;

在CPU的取指地址计算电路中增加:

软件可配置寄存器,分为两部分,一部分存储用于比较的源地址低位值;另一部分存储用于做加法运算的目的地址偏移量;

低位地址比较电路,在CPU运行的每个时钟周期,将软件可配置寄存器中存储的源地址低位值,与CPU的当前取指地址的等位宽的低位部分进行比较,并将比较结果输出给地址路选器;

地址加法电路,在CPU运行的每个时钟周期,将软件可配置寄存器中存储的目的地址偏移量,与CPU的当前取指地址进行加法运算,并将结果输出给地址路选器;

地址路选电路,在CPU运行的每个时钟周期,接收低位地址比较电路的比较结果,如果比较结果为相等,则将地址加法电路的运算结果路选输出,并在下个时钟周期,将当前取指地址更新为地址加法电路的运算结果;如果比较结果为不等,则将CPU计算出的下一个取指地址路选输出。

2.根据权利要求1所述的一种CPU中加速无条件跳转的方法,其特征在于:所述软件可配置寄存器可以通过CPU的move指令或类似指令进行赋值。

3.一种实现权利要求1所述的CPU中加速无条件跳转的方法的CPU模块。

一种CPU中加速无条件跳转的方法及电路

技术领域

[0001] 本发明属于集成电路中的嵌入式处理器技术领域，尤其是涉及一种CPU中加速无条件跳转的方法及电路。

背景技术

[0002] CPU(中央处理器)的核心功能是读取并执行软件程序指令。读取并执行指令的第一步就是取指，即CPU向总线或存储设备发出目标指令的地址，总线或存储设备根据CPU发出的地址，返回目标指令给CPU的过程。程序一般是顺序读取并执行的，直到发生程序的跳转。也就是说，取指时目标指令的地址是顺序递增的，直到遇到跳转指令。

[0003] 从CPU发出地址，到总线或存储设备返回目标指令，是有一定延迟时间的。为了整体上在一定时间内尽量多的取指，CPU通常采用流水线的方式发出地址和接受指令，如图1所示。CPU在时间t0发出目标地址0，在下一个cycle(时钟周期)不等指令0返回，就继续顺序发出地址1，如此下去。当总线或存储设备返回目标地址0所对应的指令0时，CPU已流水线化地发出了d个地址，即CPU的取指延迟是d。

[0004] 如果指令0经过CPU译码，发现正好是一条跳转指令，它的执行会将CPU的下一条指令的目标地址跳转到了一个新的地址n，那么时间t(d+j)之前发出的地址和已经取得的指令，就都作废了。CPU重新流水线化地发出地址n及其后续地址。当指令n进入CPU时，时间已经过去了t(d+j+d)了。

[0005] 总结起来，一条跳转指令，使得CPU相对浪费了d+j个cycle，其中d是取指延迟时间，j是跳转指令译码和目标地址的计算时间。程序的跳转打断了原有的取指流水线，不仅使CPU陷入等待而不能全速运行，而且取到的很多作废的指令数据，白白浪费了宝贵的总线带宽资源。可以说，程序跳转是CPU性能的主要影响因素之一。

[0006] 程序跳转，按照类型，可以分为无条件跳转和条件跳转两大类。无条件跳转是指程序会无条件地跳转到另一个确定的指令地址。在软件上通常对应着子函数的调用和退出。条件跳转是指程序会根据某个变量的值，来判断是否跳转到另一个确定的指令地址。由于条件跳转不在本文讨论范围内，因此本文之后的跳转都默认指代无条件跳转。

[0007] 对无条件跳转的加速方法可以分为软件和硬件两个方面。在软件上着重减少程序跳转的次数，比如优化编译器，采用内联函数等。在硬件上，着重减少跳转产生的延迟开销，比如采用跳转地址查找表等。

[0008] 跳转地址查找表，其基本结构与CPU中常用的缓存(Cache)基本一致。CPU中的取指单元，流水线化地向总线发出指令的目的地址，每条地址除了送至总线外，还送入跳转地址查找表中，与各表项中存储的源地址进行逐一比较。如果与某个源地址一致(即所谓的命中)，该源地址对应的目的地址，就被路选出来，作为下一条取指地址。上述过程使得取指流水线无缝地跳转到了新的程序段继续运行，避免了跳转的延迟开销，其效果如图2所示。

[0009] 图2中在t0时刻地址0在跳转地址查找表中命中，查得新的指令地址为n，因此在t1时刻CPU直接向总线发出地址n，并在t2时刻继续顺序寻址n+1。经过td延迟后，总线依次流

水化地返回指令0,指令n,指令n+1等。t0时刻的无缝跳转切换,节省了d+j个cycle。指令n在译码执行时发现需要跳转至地址f,即t1时刻的地址n未在查找表中命中,导致在t(d+j+1)时刻CPU重新发出地址f,并等待至t(2d+j+1)时刻才取得指令f,浪费了d+j个cycle。

[0010] 由上可知,跳转地址查找表对跳转的加速效果是十分理想的,但前提是跳转源地址在查表时必须命中。

[0011] 但在实际情况中,由于采用Cache结构,源地址到目的地址的首次跳转,都是未命中的。只有再次发生相同的跳转时才可能命中,因此对于程序中只发生一次的那些跳转,查找表并无作用。

[0012] 此外,由于查找表容量有限,记录的跳转只有最近发生的几个,在此之前的跳转,即使已经发生过,也无法命中。因此对于程序中多次发生,但在时间上并不集中的跳转,查找表也作用不大。

发明内容

[0013] 有鉴于此,本发明旨在提出一种CPU中加速无条件跳转的方法及电路,以解决上述问题。

[0014] 本发明的核心思想是:通过在CPU中加入软件可寻址寄存器及相应的取指计算与控制电路,从而实现通过寄存器访问指令,直接无缝切换CPU取指流水线,达到在绝大多数情况下加速无条件跳转的效果。

[0015] 为达到上述目的,本发明的技术方案是这样实现的:

[0016] 第一方面,本发明提供一种CPU中加速无条件跳转的方法,包括如下步骤:

[0017] 1)首先存储源地址低位值和目的地址偏移量;

[0018] 2)在CPU运行的每个时钟周期,将存储的源地址低位值与CPU的当前取指地址的等位宽的低位部分进行比较得到比较结果;

[0019] 3)在CPU运行的每个时钟周期,将目的地址偏移量与CPU的当前取指地址进行加法运算得到相加结果;

[0020] 4)在CPU运行的每个时钟周期,判断比较结果是否为相等:

[0021] 若相等,则将步骤3)得到的相加结果作为最终地址进行输出,并在下个时钟周期,将当前取指地址更新为相加结果;

[0022] 若不等,则将CPU计算出的下一个取指地址作为最终地址进行输出。

[0023] 第二方面,本发明提供一种CPU中加速无条件跳转的电路,在CPU的取指地址计算电路中增加:

[0024] 软件可配置寄存器,分为两部分,一部分存储用于比较的源地址低位值;另一部分存储用于做加法运算的目的地址偏移量;

[0025] 低位地址比较电路,在CPU运行的每个时钟周期,将软件可配置寄存器中存储的源地址低位值,与CPU的当前取指地址的等位宽的低位部分进行比较,并将比较结果输出给地址路选器;

[0026] 地址加法电路,在CPU运行的每个时钟周期,将软件可配置寄存器中存储的目的地址偏移量,与CPU的当前取指地址进行加法运算,并将结果输出给地址路选器;

[0027] 地址路选电路,在CPU运行的每个时钟周期,接收低位地址比较电路的比较结果,

如果比较结果为相等,则将地址加法电路的运算结果路选输出,并在下个时钟周期,将当前取指地址更新为地址加法电路的运算结果;如果比较结果为不等,则将CPU计算出的下一个取指地址路选输出。

[0028] 相对于现有技术,本发明所述的方法及电路具有以下优势:

[0029] 本发明实现通过软件指令,直接无缝切换CPU取指流水线,达到在绝大多数情况下加速无条件跳转的效果。

附图说明

[0030] 构成本发明的一部分的附图用来提供对本发明的进一步理解,本发明的示意性实施例及其说明用于解释本发明,并不构成对本发明的不当限定。在附图中:

[0031] 图1为现有的无跳转加速的CPU进行跳转取指时的流水线示意图;

[0032] 图2为现有的带有跳转预测的CPU进行跳转取指时的流水线示意图;

[0033] 图3为本发明创造的CPU中加速无条件跳转的电路的原理框图。

具体实施方式

[0034] 需要说明的是,在不冲突的情况下,本发明中的实施例及实施例中的特征可以相互组合。

[0035] 需要说明的是,在本文中,术语“包括”、“包含”或者其任何其他变体意在涵盖非排他性的包含,从而使得包括一系列要素的过程、方法、物品或者设备不仅包括那些要素,而且还包括没有明确列出的其他要素,或者是还包括为这种过程、方法、物品或者设备所固有的要素。在没有更多限制的情况下,由语句“包括……”限定的要素,并不排除在包括所述要素的过程、方法、物品或者设备中还存在另外的相同要素。下面将参考附图并结合实施例来详细说明本发明。

[0036] 本发明的一种CPU中加速无条件跳转的方法,包括如下步骤:

[0037] 1)首先存储源地址低位值和目的地址偏移量;

[0038] 2)在CPU运行的每个时钟周期,将存储的源地址低位值与CPU的当前取指地址的等位宽的低位部分进行比较得到比较结果;

[0039] 3)在CPU运行的每个时钟周期,将目的地址偏移量与CPU的当前取指地址进行加法运算得到相加结果;

[0040] 4)在CPU运行的每个时钟周期,判断比较结果是否为相等:

[0041] 若相等,则将步骤3)得到的相加结果作为最终地址进行输出,并在下个时钟周期,将当前取指地址更新为相加结果;

[0042] 若不等,则将CPU计算出的下一个取指地址作为最终地址进行输出。

[0043] 本发明实现上述CPU中加速无条件跳转的方法的电路结构,如图3所示,在CPU的取指地址计算电路中增加:

[0044] 软件可配置寄存器,可以分为两部分,一部分存储着用于比较的源地址的低位地址值;另一部分存储着用于做加法运算的目的地址偏移量。该寄存器可以通过CPU的move指令或类似指令,进行赋值;

[0045] 低位地址比较电路,在CPU运行的每个时钟周期,将软件可配置寄存器中存储的源

地址低位值，与CPU的当前取指地址的等位宽的低位部分进行比较，并将比较结果输出给地址路选器；

[0046] 地址加法电路，在CPU运行的每个时钟周期，将软件可配置寄存器中存储的目的地址偏移量，与CPU的当前取指地址进行加法运算，并将结果输出给地址路选器；

[0047] 地址路选电路，在CPU运行的每个时钟周期，接收低位地址比较电路的比较结果，如果比较结果为相等，则将地址加法电路的运算结果路选输出，并在下个时钟周期，将当前取指地址更新为地址加法电路的运算结果；如果比较结果为不等，则将CPU计算出的下一个取指地址路选输出。

[0048] 本发明提出了一种使用在嵌入式CPU中的，用于加速程序中无条件跳转的方法及其电路实现。该方法通过在CPU中加入软件可寻址寄存器及相应的地址计算与控制电路，从而实现通过软件指令，直接无缝切换CPU取指流水线，达到在绝大多数情况下加速无条件跳转的效果。

[0049] 以上所述仅为本发明的较佳实施例而已，并不用以限制本发明，凡在本发明的精神和原则之内，所作的任何修改、等同替换、改进等，均应包含在本发明的保护范围之内。

时间：	t0	t1	…	td	t（d+1）	…	t（d+j）	…	t（2d）	…	t（2d+j）
CPU 发出的地址：	地址 0	地址 1	…	地址 d	地址 d+1	…	地址 n	…	…	…	地址 n+d
总线返回的指令：				指令 0	指令 1	…	…	…	指令 d	…	指令 n

图1

时间：	t0	t1	t2	…	td	t（d+1）	t（d+2）	…	t（d+j+1）	…	t（2d+j+1）
CPU 发出的地址：	地址 0	地址 n	地址 n+1	…	…	…	…		地址 f	…	地址 f+d
总线返回的指令：					指令 0	指令 n	指令 n+1	…	…	…	指令 f

图2

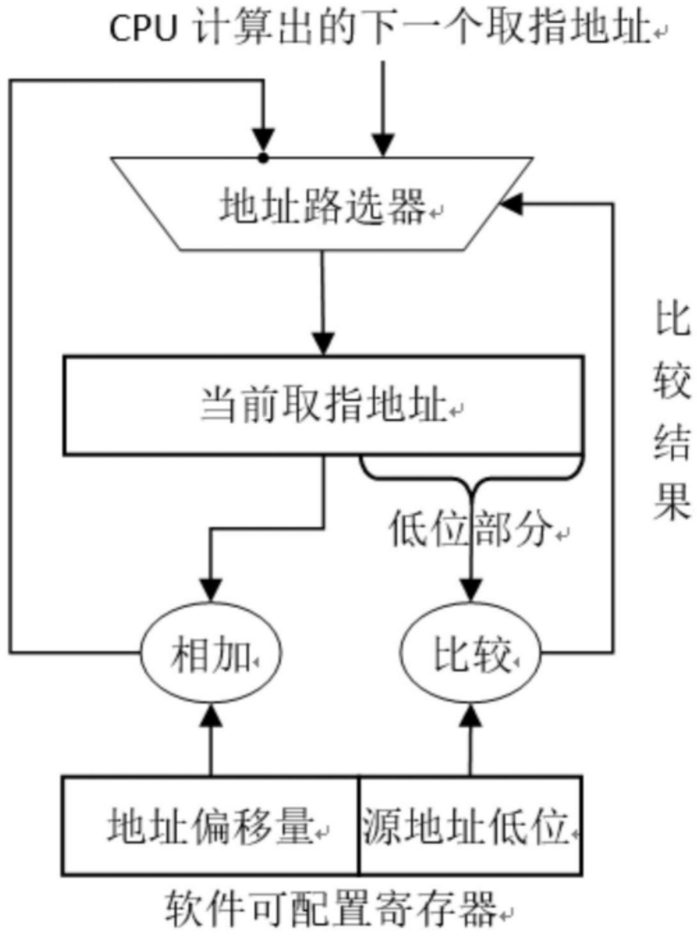

CPU 计算出的下一个取指地址
地址路选器
比较结果
当前取指地址
低位部分
相加
比较
地址偏移量
源地址低位
软件可配置寄存器

图3