(12) 按照专利合作条约所公布的国际申请

(19) 世界知识产权组织
国 际 局

(43) 国际公布日
2021 年 2 月 18 日 (18.02.2021)

WIPO | PCT

(10) 国际公布号
WO 2021/027253 A1

(51) 国际专利分类号:
G06F 9/30 (2006.01)

(21) 国际申请号: PCT/CN2020/071423

(22) 国际申请日: 2020 年 1 月 10 日 (10.01.2020)

(25) 申请语言: 中文

(26) 公布语言: 中文

(30) 优先权:
201910739180.8 2019年8月12日 (12.08.2019) CN

(71) 申请人: 创新先进技术有限公司 (**ADVANCED NEW TECHNOLOGIES CO., LTD.**) [—/CN]; 开曼群岛大开曼岛乔治镇医院路27号开曼企业中心, Grand Cayman KY1-9008 (KY)。

(72) 发明人: 马凌 (**MA, Ling**); 中国浙江省杭州市余杭区文一西路969号3号楼5楼阿里巴巴集团法务部, Zhejiang 311121 (CN)。 周伟 (**ZHOU, Wei**); 中国浙江省杭州市余杭区文一西路969号3号楼5楼阿里巴巴集团法务部, Zhejiang 311121 (CN)。 何昌华 (**HE, Changhua**); 中国浙江省杭州市余杭区文一西路969号3号楼5楼阿里巴巴集团法务部, Zhejiang 311121 (CN)。

(74) 代理人: 北京博思佳知识产权代理有限公司 (**BEIJING BESTIPR INTELLECTUAL PROPERTY LAW CORPORATION**); 中国北京市海淀区上地三街9号嘉华大厦B座409, Beijing 100085 (CN)。

(81) 指定国 (除另有指明, 要求每一种可提供的国家保护): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW。

(54) **Title:** METHOD, APPARATUS AND SYSTEM FOR MULTITHREADED PROCESSING

(54) **发明名称:** 用于多线程处理的方法、装置及系统

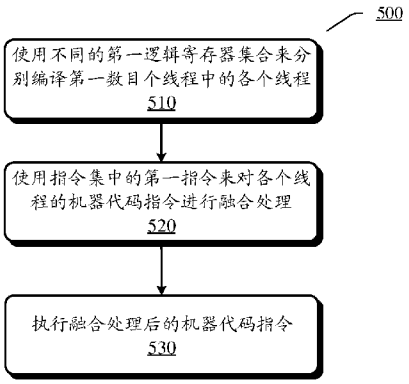


图 5

510 Respectively compile each of a first number of threads by using different first logic register sets

520 Carry out fusion processing on machine instruction codes of the threads by using a first instruction in an instruction set

530 Execute the machine instruction codes after fusion processing

(57) **Abstract:** Provided in the embodiments are a method and apparatus for multithreaded processing. In the method, for each of a first number of threads to be processed, compiling is carried out by using different first logic register sets, so as to obtain a corresponding machine instruction code; and fusion processing is carried out on the machine instruction codes of the threads by using a first instruction in an instruction set supported by a processing core, so as to obtain a machine instruction code of a single fused thread, wherein the first instruction comprises an effective address fetch instruction and a transfer instruction, the effective address fetch instruction and the transfer instruction are obtained through compiling by using a second logic register different from the first logic register set, and the transfer instruction is set in the machine instruction codes corresponding to the threads and is used for realizing a jump operation between the machine instruction codes of the threads.

WO 2021/027253 A1

[见续页]

(84) 指定国(除另有指明，要求每一种可提供的地区保护)：ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), 欧亚 (AM, AZ, BY, KG, KZ, RU, TJ, TM), 欧洲 (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG)。

本国际公布:

— 包括国际检索报告(条约第21条(3))。

(57) 摘要：本说明书实施例提供一种用于多线程处理的方法及装置。在该方法中，针对待处理的第一数目个线程中的各个线程，使用不同的第一逻辑寄存器集合来进行编译以得到对应的机器指令代码；以及使用处理内核所支持的指令集中的第一指令来对各个线程的机器指令代码进行融合处理，以得到单个融合线程的机器指令代码，其中，所述第一指令包括取有效地址指令和转移指令，所述取有效地址指令和转移指令利用与第一逻辑寄存器集合不同的第二逻辑寄存器来编译得到，所述转移指令被设置在各个线程所对应的机器指令代码中，用于实现各个线程的机器指令代码之间的跳转操作。

用于多线程处理的方法、装置及系统

技术领域

[01] 本说明书实施例通常涉及计算机应用技术，更具体地，涉及用于多线程处理的方法、装置及系统。

背景技术

[02] 在多线程处理机制中，处理内核（例如，CPU）根据线程数据在每个指令周期以轮询的方式进行指令存取和解码操作，完成之后将指令放到对应的硬件指令解码队列中，同时 CPU 执行单元在每个指令周期会轮询地从每个硬件指令解码队列得到解码后的指令，然后放入 CPU 执行单元开始运行。

[03] 在上述多线程处理机制中，CPU 会为每个线程准备一个独立的硬件运行环境，以便运行该线程。例如，CPU 会为每个线程准备对应的指令预测、指令执行、指令提交等硬件资源。通常，在所运行的线程从 1 个增加到 2 个时，会增加 5%的硬件成本。多线程处理机制所运行的线程数目越多，需要增加的硬件成本也会越多。

发明内容

[04] 鉴于上述问题，本说明书实施例提供了一种用于多线程处理的方法、装置及系统。利用该方法、装置及系统，能够将多个线程融合为单个融合线程来运行，从而在无需增加硬件成本的情况下实现多线程处理。

[05] 根据本说明书实施例的一个方面，提供了一种用于多线程处理的方法，包括：针对待处理的第一数目个线程中的各个线程，使用不同的第一逻辑寄存器集合来进行编译以得到对应的机器指令代码；以及使用处理内核所支持的指令集中的第一指令来对各个线程的机器指令代码进行融合处理，以得到单个融合线程的机器指令代码，其中，所述第一指令包括取有效地址指令和转移指令，所述取有效地址指令和转移指令利用与所述第一逻辑寄存器集合不同的第二逻辑寄存器来编译得到，所述转移指令被设置在所述各个线程所对应的机器指令代码中，用于实现所述各个线程的机器指令代码之间的跳转操作。

[06] 可选地，在上述方面的一个示例中，所述第一数目不超过第一预定阈值，所述第

一预定阈值是基于所述处理内核可使用的逻辑寄存器的数目来确定的。

[07] 可选地，在上述方面的一个示例中，所述转移指令包括通过对所述各个线程的机器指令代码中的全部或部分跳转指令进行修改而获得的第一转移指令。

[08] 可选地，在上述方面的一个示例中，所述转移指令还包括基于转移指令增加规则增加的第二转移指令。

[09] 可选地，在上述方面的一个示例中，所述转移指令增加规则包括：使得所述转移指令彼此间隔不超过第二预定数目条指令;和/或使用指令编译器调优策略来在合适的机器指令之后增加所述第二转移指令。

[10] 可选地，在上述方面的一个示例中，所述第二预定数目是基于所述处理内核在单个指令周期所能执行的指令数目确定的。

[11] 可选地，在上述方面的一个示例中，经过融合后的每个线程具有基于转移指令分割出的至少一个指令跳转块。

[12] 可选地，在上述方面的一个示例中，针对除所述第一数目个线程中的第一线程之外的每个线程，所述取有效地址指令包括单条第一取有效地址指令，所有第一取有效地址指令都被放置在所述融合线程的机器指令代码的开始处，用于将该线程的第一条指令的地址传送到对应的第二逻辑寄存器，以及针对所述第一数目个线程中的每个线程中的每个指令跳转块，所述取有效地址指令包括两条第二取有效地址指令，其中，一条第二取有效地址指令用于将该指令跳转块的第一条指令的地址传送到对应的第二逻辑寄存器，另一第二取有效地址指令用于将位于该指令跳转块之后的第一条指令的地址传送到对应的第二逻辑寄存器，所述两条第二取有效地址指令被放置在各自线程的对应指令跳转块和上一指令跳转块之间，以及所述转移指令包括条件传送指令和无条件跳转指令。

[13] 可选地，在上述方面的一个示例中，所述指令集包括 leaxxj 指令，针对除所述第一数目个线程中的第一线程之外的每个线程，所述取有效地址指令包括单条第一取有效地址指令，所述第一取有效地址指令用于将该线程的第一条指令的地址传送到对应的第二逻辑寄存器，并且所有第一取有效地址指令都被放置在所述融合线程的机器指令代码的开始处，以及针对所述第一数目个线程中的各个线程中的每个指令跳转块，所述转移指令包括单条 leaxxj 指令，其中，所述 leaxxj 指令的指令格式为 leaxxj L_loop_i（rm），rn，其中，逻辑寄存器 rm 表示对应的指令跳转块所属线程的第一取有效地址指令所使用的第二逻辑寄存器，以及逻辑寄存器 rn 表示需要跳转到的线程的第一取有效地址指

令所使用的第二逻辑寄存器，所述 leaxxj 指令用于指示在条件满足时将 L_loop_i 所指向的指令地址放入逻辑寄存器 rm，在条件不满足时将所述 leaxxj 指令之后的第一条指令的指令地址放入逻辑寄存器 rm，并且跳转到逻辑寄存器 rn 中存储的地址所对应的指令。

[14] 可选地，在上述方面的一个示例中，所述第二逻辑寄存器具有用于指示该第二逻辑寄存器的使用状态的标志位，所述使用状态包括用于表示正常使用的第一状态和用于表示该逻辑寄存器指向另一逻辑寄存器的第二状态，在进行机器指令运行时，如果所述处理内核识别出 leaxxj 指令中的 L_loop_i 被赋值为指定数值，则释放第二逻辑寄存器 rn 所指向的物理寄存器，使得第二逻辑寄存器 rn 指向目标跳转线程所使用的第二逻辑寄存器，并且将第二逻辑寄存器 rn 的标志位赋值为指示第二状态。

[15] 可选地，在上述方面的一个示例中，在进行机器指令运行时，如果所述处理内核已经取回 leaxxj 指令，则将与所述 leaxxj 指令中的逻辑寄存器 rn 所存放的地址对应的指令块预取到所述处理内核的指令解码队列中。

[16] 可选地，在上述方面的一个示例中，所述方法还包括：经由所述处理内核来以指令块为单位获取所述融合线程中的机器指令来运行。

[17] 可选地，在上述方面的一个示例中，所述指令块的大小由所述处理内核在单个指令周期读取指令的能力来确定。

[18] 可选地，在上述方面的一个示例中，在所获取的指令块中包括转移指令并且该指令块的结束指令不是所述转移指令时，将该指令块调整为以所述转移指令为结束指令。

[19] 根据本说明书实施例的另一方面，提供一种用于多线程处理的装置，包括：线程编译单元，针对待处理的第一数目个线程中的各个线程，使用不同的第一逻辑寄存器集合来进行编译以得到对应的机器指令代码；以及线程融合单元，使用处理内核所支持的指令集中的第一指令来对各个线程的机器指令代码进行融合处理，以得到单个融合线程的机器指令代码，其中，所述第一指令包括取有效地址指令和转移指令，所述取有效地址指令和转移指令利用与所述第一逻辑寄存器集合不同的第二逻辑寄存器来编译得到，所述转移指令被设置在所述各个线程所对应的机器指令代码中，用于实现所述各个线程的机器指令代码之间的跳转操作。

[20] 可选地，在上述方面的一个示例中，所述装置还包括：转移指令获得单元，通过对所述各个线程的机器指令代码中的全部或部分跳转指令进行修改而获得的第一转移指令。

[21] 可选地，在上述方面的一个示例中，所述转移指令获得单元进一步基于转移指令增加规则增加第二转移指令。

[22] 可选地，在上述方面的一个示例中，所述转移指令增加规则包括：使得所述转移指令彼此间隔不超过第二预定数目条指令;和/或使用指令编译器调优策略来在合适的机器指令之后增加所述第二转移指令。

[23] 可选地，在上述方面的一个示例中，经过融合后的每个线程具有基于转移指令分割出的至少一个指令跳转块。

[24] 可选地，在上述方面的一个示例中，针对除所述第一数目个线程中的第一线程之外的每个线程，所述取有效地址指令包括单条第一取有效地址指令，所有第一取有效地址指令都被放置在所述融合线程的机器指令代码的开始处，用于将该线程的第一条指令的地址传送到对应的第二逻辑寄存器，以及针对所述第一数目个线程中的每个线程中的每个指令跳转块，所述取有效地址指令包括两条第二取有效地址指令，其中，一条第二取有效地址指令用于将该指令跳转块的第一条指令的地址传送到对应的第二逻辑寄存器，另一第二取有效地址指令用于将位于该指令跳转块之后的第一条指令的地址传送到对应的第二逻辑寄存器，所述两条第二取有效地址指令被放置在各自线程的对应指令跳转块和上一指令跳转块之间，以及所述转移指令包括条件传送指令和无条件跳转指令。

[25] 可选地，在上述方面的一个示例中，所述指令集包括 leaxxj 指令，针对除所述第一数目个线程中的第一线程之外的每个线程，所述取有效地址指令包括单条第一取有效地址指令，所述第一取有效地址指令用于将该线程的第一条指令的地址传送到对应的第二逻辑寄存器，并且所有第一取有效地址指令都被放置在所述融合线程的机器指令代码的开始处，以及针对所述第一数目个线程中的各个线程中的每个指令跳转块，所述转移指令包括单条 leaxxj 指令，其中，所述 leaxxj 指令的指令格式为 leaxxj L_loop_i（rm），rn，其中，逻辑寄存器 rm 表示对应的指令跳转块所属线程的第一取有效地址指令所使用的第二逻辑寄存器，以及逻辑寄存器 rn 表示需要跳转到的线程的第一取有效地址指令所使用的第二逻辑寄存器，所述 leaxxj 指令用于指示在条件满足时将 L_loop_i 指向的指令地址放入逻辑寄存器 rm，在条件不满足时将所述 leaxxj 指令之后的第一条指令的指令地址放入逻辑寄存器 rm，并且跳转到逻辑寄存器 rn 中存储的地址所对应的指令。

[26] 根据本说明书实施例的另一方面，提供一种用于多线程处理的系统，包括：编译器，包括如上所述的用于多线程处理的装置，将第一数目个线程编译为单个融合线程的机器指令代码；指令集；以及处理内核，以指令块为单位获取所述融合线程中的机器指

令来运行。

[27] 可选地，在上述方面的一个示例中，所述指令集包括 leaxxj 指令，所述 leaxxj 指令的指令格式为 leaxxj L_loop_i（rm），rn，其中，逻辑寄存器 rm 表示对应的指令跳转块所属线程的第一取有效地址指令所使用的第二逻辑寄存器，以及逻辑寄存器 rn 表示需要跳转到的线程的第一取有效地址指令所使用的第二逻辑寄存器，所述 leaxxj 指令用于指示在条件满足时将 L_loop_i 所指向的指令地址放入逻辑寄存器 rm，在条件不满足时将所述 leaxxj 指令之后的第一条指令的指令地址放入逻辑寄存器 rm，并且跳转到逻辑寄存器 rn 中存储的地址所对应的指令。

[28] 可选地，在上述方面的一个示例中，所述第二逻辑寄存器具有用于指示该第二逻辑寄存器的使用状态的标志位，所述使用状态包括用于表示正常使用的第一状态和用于表示该逻辑寄存器指向另一逻辑寄存器的第二状态，在进行机器指令运行时，如果所述处理内核识别出 leaxxj 指令中的 L_loop_i 被赋值为指定数值，则所述处理内核释放第二逻辑寄存器 rn 所指向的物理寄存器，使得第二逻辑寄存器 rn 指向目标跳转线程所使用的第二逻辑寄存器，并且将第二逻辑寄存器 rn 的标志位赋值为指示第二状态。

[29] 可选地，在上述方面的一个示例中，在进行机器指令运行时，如果所述处理内核已经取回 leaxxj 指令，则所述处理内核将与所述 leaxxj 指令中的逻辑寄存器 rn 所存放的地址对应的指令块预取到所述处理内核的指令解码队列中。

[30] 根据本说明书实施例的又一方面，提供一种计算设备，包括：一个或多个处理器，以及与所述一个或多个处理器耦合的存储器，所述存储器存储指令，当所述指令被所述一个或多个处理器执行时，使得所述至少一个处理器执行如上所述的用于多线程处理的方法。

[31] 根据本说明书实施例的再一方面，提供一种机器可读存储介质，其存储有可执行指令，所述指令当被执行时使得所述机器执行如上所述的用于多线程处理的方法。

附图说明

[32] 通过参照下面的附图，可以实现对于本说明书实施例的内容的本质和优点的进一步理解。在附图中，类似组件或特征可以具有相同的附图标记。

[33] 图 1 示出了现有技术中的多线程处理过程的示意图；

[34] 图 2 示出了现有技术中的用于对多个线程进行编译处理的示例示意图；

[35] 图 3 示出了指令运行过程的示例示意图；

[36] 图 4 示出了根据本说明书的实施例的多线程处理系统的方框图；

[37] 图 5 示出了根据本说明书的实施例的用于多线程处理的方法流程图；

[38] 图 6 示出了根据本说明书的实施例的线程融合处理的一个示例示意图；

[39] 图 7 示出了根据本说明书的实施例的线程融合处理的另一示例示意图；

[40] 图 8 示出了根据本说明书的实施例的融合线程运行过程的流程图；

[41] 图 9 示出了根据本说明书的实施例的线程退出处理的线程示例示意图；

[42] 图 10 示出了根据本说明书的实施例的多线程编译装置的方框图；

[43] 图 11 示出了根据本说明书的实施例的用于多线程处理的计算设备的方框图。

具体实施方式

[44] 现在将参考示例实施方式讨论本文描述的主题。应该理解，讨论这些实施方式只是为了使得本领域技术人员能够更好地理解从而实现本文描述的主题，并非是对权利要求书中所阐述的保护范围、适用性或者示例的限制。可以在不脱离本说明书实施例的内容的保护范围的情况下，对所讨论的元素的功能和排列进行改变。各个示例可以根据需要，省略、替代或者添加各种过程或组件。例如，所描述的方法可以按照与所描述的顺序不同的顺序来执行，以及各个步骤可以被添加、省略或者组合。另外，相对一些示例所描述的特征在其它例子中也可以进行组合。

[45] 如本文中使用的，术语“包括”及其变型表示开放的术语，含义是“包括但不限于”。术语“基于”表示“至少部分地基于”。术语“一个实施例”和“一实施例”表示“至少一个实施例”。术语“另一个实施例”表示“至少一个其他实施例”。术语“第一”、“第二”等可以指代不同的或相同的对象。下面可以包括其他的定义，无论是明确的还是隐含的。除非上下文中明确地指明，否则一个术语的定义在整个说明书中是一致的。

[46] 图 1 示出了现有技术中的多线程处理过程 100 的示意图。

[47] 如图 1 所示，多线程处理过程 100 可以包括线程编译过程 110 和线程运行过程 120。

[48] 具体地，在线程编译过程 110 中，针对待处理的多个线程中的各个线程，使用相同的逻辑寄存器集合来进行编译以得到对应的机器指令代码。这里，逻辑寄存器是指在

程序代码中使用的寄存器，并且逻辑寄存器指向物理寄存器。

[49] 图 2 示出了现有技术中的用于对多个线程进行编译处理的示例示意图。在图 2 中，示出了 3 个线程 Thread1、Thread2 和 Thread3。如图 2 所示，Thread1、Thread2 和 Thread3 使用了相同的逻辑寄存器集合{r1，r2，r3，r4，……}来进行编译。其中，线程 Thread1 使用了逻辑寄存器 r1-r3，线程 Thread2 使用了逻辑寄存器 r1 和 r3，以及线程 Thread3 使用逻辑寄存器 r1，r2 和 r4。

[50] 在本说明书的实施例中，每条机器指令包括操作码和操作数，其中，操作码用于指明指令所要完成操作的性质和功能，比如，add，mul，mov，sub，jmp 等。操作数用于指明操作码执行时的操作对象。操作数的形式可以是数据本身.也可以是存放数据的内存单元地址或寄存器名称。操作数又分为源操作数和目的操作数，源操作数指明参加运算的操作数来源，目的操作数地址指明保存运算结果的存储单元地址或寄存器名称。

[51] 此外，处理内核（例如，中央处理器 CPU）还为多个线程中的每个线程创建独立的线程运行环境。所述线程运行环境包括用于执行指令预测的硬件资源、用于执行指令执行的硬件资源，用于执行指令提交的硬件资源等。另外，还为多个线程中的每个线程创建各自的硬件指令解码队列。

[52] 在线程运行过程 120 中，CPU 根据线程数据在每个指令周期以轮询的方式进行指令存取和解码操作，完成之后将指令放到各自的硬件指令解码队列中。同时，CPU 执行单元在每个指令周期会轮询地从每个硬件指令解码队列得到解码后的指令，然后放入 CPU 执行单元开始运行。

[53] 图 3 示出了指令运行过程的示例示意图。

[54] 如图 3 所示，在块 310，执行指令获取（Instruction Fetch，IF）操作。指令获取操作是指将待运行的指令从主存中取到指令寄存器中。程序计数器 PC 中的数值用来指示当前指令在主存中的位置。当一条指令被取出后，PC 中的数值将根据指令字长度而自动递增：若为单字长指令，则增加 1 个字节；若为双字长指令，则增加 2 个字节，依此类推。

[55] 在取出待运行指令后，在块 320，执行指令译码（Instruction Decode，ID）操作。在指令译码操作中，指令译码器按照预定的指令格式，对取回的指令进行拆分和解释，识别区分出不同的指令类别以及各种获取操作数的方法。在组合逻辑控制的计算机中，指令译码器对不同的指令操作码产生不同的控制电位，以形成不同的微操作序列。在微

程序控制的计算机中，指令译码器用指令操作码来找到执行该指令的微程序的入口，并从此入口开始执行。

[56] 在完成指令译码操作后，在块 330，进行指令执行（Execution， EX）操作。在指令执行操作中，完成指令所规定的各种操作来实现指令功能。为此，CPU 的不同组件部分被连接起来，以执行所需操作。例如，如果要求完成加法运算，则算术逻辑单元 ALU 将被连接到一组输入和一组输出，输入端提供需要相加的数值，输出端将含有最后的运算结果。

[57] 此外，可选地，在块 340，根据指令需要，执行操作数存取操作。在操作数存取操作中，根据指令地址码得到操作数在主存中的地址，并从主存中读取该操作数以用于运算。

[58] 在完成指令运行后，在块 350，执行结果写回（Writeback，WB）操作。在结果回写操作中，将指令执行操作的运行结果数据“写回”到某种存储单元中。例如，结果数据经常被写到 CPU 的内部寄存器中，以便被后续的指令快速地存取。在有些情况下，结果数据也可以被写入相对较慢、但较廉价且容量较大的主存。许多指令还会改变程序状态字寄存器中标志位的状态，这些标志位用于标识不同的操作结果，可被使用来影响程序动作。

[59] 在指令执行完毕并且完成结果数据写回操作后，若无意外事件（例如，结果溢出等）发生，处理内核就从程序计数器 PC 中取得下一条指令地址，开始新一轮的指令运行过程。

[60] 图 3 中示出的是单个线程中的指令运行过程的示意图。在多线程处理机制中，操作系统使用线程调度机制来在多个线程之间进行线程切换。例如，操作系统为 CPU 允许并行执行的多个线程中的每个线程分配一个时间片，各个线程在各自的时间片内占用 CPU 来执行指令运行操作。当前线程的时间片使用完毕后就会处于就绪状态，并且让出 CPU 给其它线程占用。此外，如果线程的任务在自己的时间片结束前被阻塞或结束，则会马上进行线程切换，而不会造成 CPU 资源浪费。

[61] 此外，还可以使用超线程机制来实现多线程处理。超线程机制是指在一个线程执行整数指令集时，其它线程所执行的任务是由与执行当前线程的任务的执行单元不同的执行单元执行的任务，比如，浮点指令集。在这种情况下，可以同时执行这两个线程。超线程技术允许两个线程同时不冲突地使用 CPU 中的资源。

[62] 在上面的多线程处理方案中，CPU 需要为每个线程准备一个独立的硬件运行环境来运行该线程，从而需要更多的硬件成本。多线程处理机制所运行的线程数目越多，需要增加的硬件成本也会越多。在这种多任务处理方案中，并行处理的线程数目受限于用于线程处理的硬件条件。

[63] 为了解决上述问题，本说明书实施例提供了一种多线程处理方案。在该多线程处理方案中，通过使用不同的寄存器集合来分别编译多个线程，并且使用处理内核所支持的指令集中的取有效地址指令和转移指令来将多个线程融合为单个融合线程，然后基于所得到的融合线程来运行，由此实现多线程处理。在这种情况下，仅仅需要提供单个线程运行所需的硬件成本，从而能够在无需增加硬件成本的情况下实现多线程处理。

[64] 图 4 示出了根据本说明书的实施例的多线程处理系统 400 的方框图。如图 4 所示，多线程处理系统 400 包括多线程编译装置 410、指令集 420 和处理内核 430。

[65] 多线程编译装置 410 被配置为使用不同的第一逻辑寄存器集合来分别编译待处理的第一数目个线程中的各个线程，以得到各个线程的机器指令代码，并且使用处理内核所支持的指令集 420 中的第一指令来对各个线程的机器指令代码进行融合处理，以得到单个融合线程的机器指令代码。所得到的单个融合线程的机器指令代码被处理内核 430 获取来进行指令运行。这里，所述第一指令包括取有效地址指令（Lea 指令）和转移指令，所述取有效地址指令和转移指令利用与第一逻辑寄存器集合不同的第二逻辑寄存器来编译得到，所述转移指令被设置在各个线程所对应的机器指令代码中，用于实现各个线程的机器指令代码之间的跳转操作。多线程编译装置 410 的操作和结构将在下面分别参照图 5-图 7 和图 9 详细描述。

[66] 这里要说明的是，第一逻辑寄存器集合和第二逻辑寄存器都属于处理内核 430 可使用的逻辑寄存器。在本说明书的实施例中，可以通过对处理内核 430 可使用的逻辑寄存器进行分割处理来得到第一逻辑寄存器集合和第二逻辑寄存器。

[67] 处理内核 430 被配置为以指令块为单位获取融合线程中的机器指令来运行。在本说明书实施例的一个示例中，指令块的大小可以由处理内核在单个指令周期读取指令的能力来确定。此外，在所获取的指令块中包括转移指令并且该指令块的结束指令不是转移指令时，可以将该指令块调整为以转移指令为结束指令。处理内核 430 的操作将在下面参照图 8 详细描述。

[68] 图 5 示出了根据本说明书的实施例的用于多线程处理的方法流程图。

[69] 如图 5 所示，在块 510，针对待处理的第一数目个线程中的各个线程，使用不同的第一逻辑寄存器集合来进行编译以得到对应的机器指令代码。在本说明书的实施例中，逻辑寄存器集合不同是指逻辑寄存器集合中的包括的所有逻辑寄存器都不同。此外，逻辑寄存器集合是通过对处理内核可使用的逻辑寄存器进行分割而得到的。例如，假设处理内核可使用的逻辑寄存器包括逻辑寄存器 r1-r10，则逻辑寄存器集合是通过对逻辑寄存器 r1-r10 进行分割而得到的。在本说明书实施例的一个示例中，第一数目被设置为不超过第一预定阈值。所述第一预定阈值是基于处理内核可使用的逻辑寄存器的数目来确定的。例如，在一个示例中，可以基于处理内核可使用的逻辑寄存器的数目以及待处理的线程所需逻辑寄存器的数目确定所述第一预定阈值。此外，在本说明书实施例的另一示例中，第一数目可以通过软件来任意设定。在这种情况下，如果所设定的第一数目超过处理内核可使用的逻辑寄存器的数目或者第一数目个线程所需逻辑寄存器的数目超过处理内核可使用的逻辑寄存器的数目，则可以使用压栈的方式来实现线程编译。

[70] 图 6 示出了根据本说明书的实施例的线程融合处理的一个示例示意图；以及图 7 示出了根据本说明书的实施例的线程融合处理的另一示例示意图。图 6 和图 7 中示出的是对图 2 中示出的线程 Thread1、Thread2 和 Thread3 的线程融合处理。在图 6 中，使用逻辑寄存器 r1-r3 组成的第一逻辑寄存器集合来对线程 Thread1 进行编译，使用逻辑寄存器 r8-r9 组成的第一逻辑寄存器集合来对线程 Thread2 进行编译，以及使用逻辑寄存器 r12-r14 组成的第一逻辑寄存器集合来对线程 Thread3 进行编译。在图 7 中，使用逻辑寄存器 r1-r3 组成的第一逻辑寄存器集合来对线程 Thread1 进行编译，使用逻辑寄存器 r8-r9 组成的第一逻辑寄存器集合来对线程 Thread2 进行编译，以及使用逻辑寄存器 r11-r13 组成的第一逻辑寄存器集合来对线程 Thread3 进行编译。

[71] 在如上得到各个线程的机器指令代码后，在块 520，使用处理内核所支持的指令集 420 中的第一指令来对各个线程的机器指令代码进行融合处理，以得到单个融合线程的机器指令代码。这里，所述第一指令包括取有效地址指令（Lea 指令）和转移指令，所述取有效地址指令和转移指令利用与所述第一逻辑寄存器集合不同的第二逻辑寄存器来编译得到，所述转移指令被设置在所述各个线程所对应的机器指令代码中，用于实现所述各个线程的机器指令代码之间的跳转操作。

[72] 取有效地址指令也称为 LEA 指令，是微机 8086/8088 系列的一条指令。LEA 指令的指令格式为：LEA 操作数 A，操作数 B。LEA 指令的指令功能：将操作数 B 的有效地址传送到操作数 A 所在的单元。这里，操作数 A 是目的操作数，以及操作数 B 是源

操作数。LEA 指令要求源操作数 B 必须是存储单元，而且目的操作数 A 必须是寄存器。

[73] 例如，在图 6 所示的方案中，针对线程 Thread1，使用逻辑寄存器 r4 和 r5 作为第二逻辑寄存器来编译与线程 Thread1 相关的第一指令，针对线程 Thread2，使用逻辑寄存器 r6 和 r7 作为第二逻辑寄存器来编译与线程 Thread2 相关的第一指令，以及针对线程 Thread3，使用逻辑寄存器 r10 和 r11 作为第二逻辑寄存器来编译与线程 Thread3 相关的第一指令。在图 7 中示出的示例中，针对线程 Thread1，使用逻辑寄存器 r4 作为第二逻辑寄存器来编译与线程 Thread1 相关的第一指令，针对线程 Thread2，使用逻辑寄存器 r6 作为第二逻辑寄存器来编译与线程 Thread2 相关的第一指令，以及针对线程 Thread3，使用逻辑寄存器 r10 作为第二逻辑寄存器来编译与线程 Thread3 相关的第一指令。关于如何使用第一指令来进行融合处理，将在下面参照图 6 和图 7 的示例来进行说明。

[74] 在一个示例中，所述转移指令可以包括通过对各个线程的机器指令代码中的全部或部分跳转指令进行修改而获得的第一转移指令。例如，在图 6 中，针对线程 Thread1，对图 2 中的线程 Thread1 中的跳转指令“1.6 ja L_loop_1”进行修改而得到的转移指令“1.10 cmovbe r5，r4”和“1.11 jmp r6”，对图 2 中的线程 Thread1 中的跳转指令“1.14 jbe L_loop_2”进行修改而得到的转移指令“1.21 cmovbe r5，r4”和“1.22 jmp r6”，但是对图 2 中的线程 Thread1 中的跳转指令“1.9 ja L_sub_again”不进行修改。针对线程 Thread2，对图 2 中的线程 Thread2 中的全部跳转指令“2.7 ja L_loop_3”进行修改而得到的转移指令“2.9 cmovbe r7，r6”和“2.10 jmp r10”。针对线程 Thread3，通过对图 2 中的线程 Thread3 中的全部跳转指令“3.6 ja L_loop_4”进行修改而得到的转移指令“3.8 cmovbe r11，r10”和“3.9 jmp r4”。在图 6 中示出的示例中，转移指令包括条件传送指令 cmovbe 和无条件跳转指令 jmp。这里要说明的是，图 6 中示出的 cmovbe 指令仅仅是示例。在本说明书实施例的其它示例中，条件传送指令可以包括 cmovle 指令，cmovnz 指令等。

[75] 在图 7 中，针对线程 Thread1，对图 2 中的线程 Thread1 中的跳转指令“1.6 ja L_loop_1”进行修改而得到转移指令“1.8 leaaj L_loop_1（r4），r6”，以及对图 2 中的线程 Thread1 中的跳转指令“1.14 jbe L_loop_2”进行修改而得到转移指令“1.16 leaaj L_loop_2（r4），r6”。同样，对图 2 中的线程 Thread1 中的跳转指令“1.9 ja L_sub_again”不进行修改。针对线程 Thread2，对图 2 中的线程 Thread2 中的跳转指令“2.7 ja L_loop_3”进行修改而得到转移指令“2.7 leaaj L_loop_3（r6），r10”。针对线程 Thread3，对图 2 中的线程 Thread3 中的跳转指令“3.6 ja L_loop_4”进行修改而得到转移指令“3.6 leaaj L_loop_4

（r10），r4”。这里，leaaj 指令是条件转移指令。关于 leaaj 指令的具体结构和含义将在下面具体描述。

[76] 在该示例中，利用各个线程的机器指令代码中的现有跳转指令来修改得到用于线程融合的转移指令，可以减少融合后的线程中的指令数目，由此提高融合线程的效率。

[77] 此外，在另一示例中，所述转移指令可以包括基于转移指令增加规则增加的第二转移指令。这里，第二转移指令不是基于各个线程中的原有指令修改得到的，而是增加的新指令。例如，假设线程 Thread1、线程 Thread2 和/或线程 Thread3 中都不包含跳转指令，或者线程 Thread1、线程 Thread2 和/或线程 Thread3 中所包含的跳转指令之间的指令间隔太大（例如，超过预定数目，比如，处理内核在单个指令周期所能执行的指令数目），则可以在对应的线程中，基于转移指令增加规则增加第二条件转移指令。

[78] 在本说明书实施例中，所述转移指令增加规则可以包括：使得转移指令彼此间隔不超过第二预定数目条指令;和/或使用指令编译器调优策略来在合适的机器指令之后增加第二转移指令。这里，所述第二预定数目可以是基于处理内核在单个指令周期所能执行的指令数目确定的。

[79] 在本说明书实施例中，“使得所述转移指令彼此间隔不超过第二预定数目条指令”是指线程中的转移指令（第一转移指令和/或第二转移指令）之间的间隔不超过第二预定数目条指令。

[80] “使用指令编译器调优策略来在合适的机器指令之后增加第二转移指令”是指通过循环测试的策略来确定可以使得编译器效率最佳的转移指令增加方式。

[81] 在该示例中，在融合前的线程中缺少跳转指令的情况下，通过按照转移指令增加规则来增加转移指令，可以使得融合后的线程能够在合适的时机进行线程跳转，由此提高融合后的线程的运行效率。

[82] 此外，在经过上述融合处理后，经过融合后的每个线程具有基于转移指令分割出的至少一个指令跳转块。例如，在图 6 中的融合方案中，针对线程 Thread1，具有 2 个指令跳转块，即，从指令“L_loop_1:1.6 add (addr1)，r1”到指令“1.11 jmp r6”的第一指令跳转块，以及从指令“L_loop_2:1.17 sub (addr1)，r4”到指令“1.22 jmp r6”的第二指令跳转块。针对线程 Thread2，具有 1 个指令跳转块，即，从指令“L_loop_3:2.5 add (addr2)，r8”到指令“2.10 jmp r10”的第三指令跳转块。针对线程 Thread3，具有 1 个指令跳转块，即，从指令“L_loop_4:3.4 sub (addr3)，r14”到指令“3.9 jmp r4”的第四

指令跳转块。

[83] 在本说明书实施例的融合处理方案的一个示例中，如图 6 中所示，针对除第一数目个线程中的第一线程之外的每个线程，取有效地址指令包括单条第一取有效地址指令，并且所有第一取有效地址指令都被放置在融合线程的机器指令代码的开始处，用于将该线程的第一条指令的地址传送到对应的第二逻辑寄存器。例如，图 6 中示出第一取有效指令“1.1 lea r6，L_Thread2(rip)”和“1.2 lea r10，L_Thread3(rip)”，其中，第一取有效指令“1.1 lea r6，L_Thread2(rip)”与线程 Thread2 对应，以及第一取有效指令“1.2 lea r10，L_Thread3(rip)”与线程 Thread3 对应。

[84] 此外，在图 6 中示出的示例中，针对第一数目个线程中的每个线程中的每个指令跳转块，取有效地址指令包括两条第二取有效地址指令，其中，一条第二取有效地址指令用于将该指令跳转块的第一条指令的地址传送到对应的第二逻辑寄存器，另一第二取有效地址指令用于将位于该指令跳转块之后的第一条指令的地址传送到对应的第二逻辑寄存器，所述两条第二取有效地址指令被放置在各自线程的对应指令跳转块和上一指令跳转块之间，以及所述转移指令包括条件传送指令和无条件跳转指令。

[85] 如图 6 中所示，针对线程 Thread1 中的第一指令跳转块，存在两条第二取有效地址指令“1.4 lea r5，Leave_loop_1(rip)”和“1.5 lea r4 L_loop_1(rip)”，即，将指令“L_loop_1: 1.6 add (addr1)，r1”（即，L_loop_1 所指向的指令）的地址传送（存放）到逻辑寄存器 r4，以及将指令“Leave_loop_1: L_sub_again 1.12 sub r1，r3”（即，Leave_loop_1 所指向的指令）的地址传送（存放）到逻辑寄存器 r5。并且该两条第二取有效地址指令被放置在指令“L_loop_1: 1.6 add (addr1)，r1”之前。针对线程 Thread1 中的第二指令跳转块，存在两条第二取有效指令“1.15 lea r5，Leave_loop_2(rip)”和“1.16 lea r4 L_loop_2(rip)”，即，将指令“L_loop_2: 1.17 sub (addr1)，r4”的地址传送（存放）到逻辑寄存器 r4，以及将指令“Leave_loop_2 ……”的地址传送（存放）到逻辑寄存器 r5。并且该两条第二取有效地址指令被放置在指令“Leave_loop_1: L_sub_again”和指令“L_loop_2: 1.17 sub (addr1)，r4”之间，即，第一指令跳转块和第二指令跳转块之间。通常，该两条第二取有效地址指令被优选放置为紧接对应的指令跳转块的第一条指令。

[86] 针对线程 Thread2 中的第三指令跳转块，存在两条第二取有效地址指令“2.1 lea r7，Leave_loop_3(rip)”和“2.2 lea r6 L_loop_3(rip)”，即，将指令“L_loop_3: 2.5 add (addr2)，r8”的地址传送（存放）到逻辑寄存器 r6，以及将指令“Leave_loop_3: 2.11 add $9，r9”的地址传送（存放）到逻辑寄存器 r7。并且该两条第二取有效地址指令被放置在指令

“L_loop_3: 2.5 add (addr2), r8”之前，即，放置在线程 Thread2 的第三指令跳转块之前。

[87] 针对线程 Thread3 中的第四指令跳转块，存在两条第二取有效地址指令“3.1 lea r11, Leave_loop_4(rip)”和“3.2 lea r10 L_loop_4(rip)”，即，将指令“L_loop_4: 3.4 sub (addr3), r14”的地址传送（存放）到逻辑寄存器 r10，以及将指令“Leave_loop_4: ……”的地址传送(存放)到逻辑寄存器 r11。并且该两条第二取有效地址指令被放置在指令“L_loop_4: 3.4 sub (addr3)，r14”之前，即，放置在线程 Thread3 的第四指令跳转块之前。

[88] 在图 6 的示例中，针对第一数目个线程中的每个线程中的每个指令跳转块，转移指令包括条件传送指令 cmovbe 和无条件跳转指令 jmp。例如，针对线程 Thread1 中的第一指令跳转块，转移指令包括条件传送指令“1.10 cmovbe r5，r4”和无条件跳转指令“1.11 jmp r6”。针对线程 Thread1 中的第二指令跳转块，转移指令包括条件传送指令“1.21 cmovbe r5，r4”和无条件跳转指令“1.22 jmp r6”。针对线程 Thread2 中的第三指令跳转块，转移指令包括条件传送指令“2.9 cmovbe r7，r6”和无条件跳转指令“2.10 jmp r10”。针对线程 Thread3 中的第四指令跳转块，转移指令包括条件传送指令“3.8 cmovbe r11，r10”和无条件跳转指令“3.9 jmp r4”。

[89] 按照图 6 中示出的线程融合方案，可以将多个线程的机器指令代码融合为单个线程的机器指令代码，由此在进行指令执行时，仅仅需要准备与单个线程对应的硬件运行环境，从而能够在无需增加硬件成本的情况下实现多线程处理。

[90] 要说明的是，在图 6 中示出的线程融合方案中，由于在跳转指令之前，处理内核还不知道跳转目的寄存器，从而需要进行寄存器预测处理。

[91] 在本说明书实施例的融合处理方案的另一示例中，如图 7 中所示，指令集 420 包括 leaxxj 指令。相应地，针对除第一数目个线程中的第一线程之外的每个线程，所述取有效地址指令包括单条第一取有效地址指令，所述第一取有效地址指令用于将该线程的第一条指令的地址传送到对应的第二逻辑寄存器。例如，图 7 中示出第一取有效指令“1.1 lea r6，L_Thread2(rip)”和“1.2 lea r10，L_Thread3(rip)”，其中，第一取有效指令“1.1 lea r6，L_Thread2(rip)”与线程 Thread2 对应，以及第一取有效指令“1.2 lea r10，L_Thread3(rip)”与线程 Thread3 对应。并且，第一取有效指令“1.1 lea r6，L_Thread2(rip)”和“1.2 lea r10，L_Thread3(rip)”被放置在融合线程的开始处。

[92] 此外，针对第一数目个线程中的各个线程中的每个指令跳转块，转移指令包括单条 leaxxj 指令。

[93] 这里，leaxxj 指令是 CPU 的指令集中的新增条件转移指令。所述 leaxxj 指令的指令格式为 leaxxj L_loop_i（rm），rn，其中，逻辑寄存器 rm 表示对应的指令跳转块所属线程的第一取有效地址指令所使用的第二逻辑寄存器，以及逻辑寄存器 rn 表示需要跳转到的线程的第一取有效地址指令所使用的第二逻辑寄存器，所述 leaxxj 指令用于指示在条件满足时将 L_loop_i 所指向的指令地址放入逻辑寄存器 rm，在条件不满足时将该 leaxxj 指令之后的第一条指令的指令地址放入逻辑寄存器 rm，并且（无条件）跳转到逻辑寄存器 rn 中存储的地址所对应的指令。根据条件判断的方式不同，leaxxj 指令可以包括 leaaj 指令，leaaej 指令，leabj 指令，leabej 指令，leagj 指令，leagej 指令，lealej 指令，lealj 指令，leanej 指令，leaej 指令，leaj 指令，leancj 指令和 leacj 指令等。

[94] 具体地，leaaj 指令的格式为 leaaj L_Loop_i(rn)，rm，用于表示如果无符号大于，则将 L_Loop_i 所指向的指令地址存入逻辑寄存器 rn，否则将当前 leaaj 指令后面的指令地址存入逻辑寄存器 rn，同时跳转到逻辑寄存器 rm。

[95] 例如：

sub r1，r2

leaaj L_Loop_i(rn)，rm

[96] 上述 leaaj 指令表示无符号比较 r2 和 r1，如果 r2 大于 r1，则将 L_Loop_i 所指向的指令地址存入 rn，否则将当前 leaaj 指令后面的指令地址存入 rn，同时跳转到 rm。

[97] leaaej 指令的格式为 leaaej L_Loop_i(rn)，rm，用于表示如果无符号大于或者等于，则将 L_Loop_i 所指向的指令地址存入逻辑寄存器 rn，否则将当前 leaaej 指令后面的指令地址存入逻辑寄存器 rn，同时跳转到逻辑寄存器 rm。

[98] 例如：

sub r1，r2

leaaej L_Loop_i(rn)，rm

[99] 上述 leaaej 指令表示无符号比较 r2 和 r1，如果 r2 大于或者等于 r1，则将 L_Loop_i 所指向的指令地址存入 rn，否则将当前 leaaej 指令后面的指令地址存入 rn，同时跳转到 rm。

[100] leabj 指令的格式为 leabj L_Loop_i(rn)，rm，用于表示如果无符号小于，则将 L_Loop_i 所指向的指令地址存入逻辑寄存器 rn，否则将当前 leabj 指令后面的指令地址

存入逻辑寄存器 rn, 同时跳转到逻辑寄存器 rm。

[101] 例如:

sub r1，r2

leabj L_Loop_i(rn)，rm

[102] 上述 leabj 指令用于表示无符号比较 r2 和 r1，如果 r2 小于 r1，则将 L_Loop_i 所指向的指令地址存入 rn，否则将当前 leabj 指令后面的指令地址存入 rn，同时跳转到 rm。

[103] leabej 指令的格式为 leabej L_Loop_i(rn)，rm，用于表示如果无符号小于或者等于，则将 L_Loop_i 所指向的指令地址存入逻辑寄存器 rn，否则将当前 leabej 指令后面的指令地址存入逻辑寄存器 rn，同时跳转到逻辑寄存器 rm。

[104] 例如:

sub r1，r2

leabej L_Loop_i(rn)，rm

[105] 上述 leabej 指令用于表示无符号比较 r2 和 r1，如果 r2 小于或者等于 r1，则将 L_Loop_i 所指向的指令地址存入 rn,否则将当前 leabej 指令后面的指令地址存入 rn，同时跳转到 rm。

[106] leagj 指令的格式为 leagj L_Loop_i(rn)，rm，用于表示有符号大于，则将 L_Loop_i 所指向的指令地址存入逻辑寄存器 rn, 否则将当前 leagj 指令后面的指令地址存入逻辑寄存器 rn, 同时跳转到逻辑寄存器 rm。

[107] 例如:

sub r1，r2

leagj L_Loop_i(rn)，rm

[108] 上述 leagj 指令用于表示有符号比较 r2 和 r1，如果 r2 大于 r1，则将 L_Loop_i 所指向的指令地址存入 rn，否则将当前 leagj 指令后面的指令地址存入 rn，同时跳转到 rm。

[109] leagej 指令的格式为 leagej L_Loop_i(rn)，rm，用于表示有符号大于或者等于，则将 L_Loop_i 所指向的指令地址存入逻辑寄存器 rn，否则将当前 leagej 指令后面的指令地址存入逻辑寄存器 rn，同时跳转到逻辑寄存器 rm。

[110] 例如:

sub r1，r2

leagej L_Loop_i(rn)，rm

[111] 上述 leagej 指令表示有符号比较 r2 和 r1，如果 r2 大于或者等于 r1，则将 L_Loop_i 所指向的指令地址存入 rn，否则将当前 leagej 指令后面的指令地址存入 rn，同时跳转到 rm。

[112] lealej 指令的格式为 lealej L_Loop_i(rn)，rm，用于表示有符号小于或者等于，则将 L_Loop_i 所指向的指令地址存入逻辑寄存器 rn，否则将当前 lealej 指令后面的指令地址存入逻辑寄存器 rn，同时跳转到逻辑寄存器 rm。

[113] 例如:

sub r1，r2

lealej L_Loop_i(rn)，rm

[114] 上述 lealej 指令表示有符号比较 r2 和 r1，如果 r2 小于或者等于 r1，则将 L_Loop_i 所指向的指令地址存入 rn，否则将当前 lealej 指令后面的指令地址存入 rn，同时跳转到 rm。

[115] lealj 指令的格式为 lealj L_Loop_i(rn)，rm，用于表示有符号小于，则将 L_Loop_i 所指向的指令地址存入逻辑寄存器 rn，否则将当前 lealj 指令后面的指令地址存入逻辑寄存器 rn，同时跳转到逻辑寄存器 rm。

[116] 例如:

sub r1，r2

lealj L_Loop_i(rn)，rm

[117] 上述 lealj 指令表示有符号比较 r2 和 r1，如果 r2 小于 r1，则将 L_Loop_i 所指向的指令地址存入 rn，否则将当前 lealj 指令后面的指令地址存入 rn，同时跳转到 rm。

[118] leanej 指令的格式为 leanej L_Loop_i(rn)，rm，用于表示不等于，则将 L_Loop_i 所指向的指令地址存入逻辑寄存器 rn，否则将当前 leanej 指令后面的指令地址存入逻辑寄存器 rn，同时跳转到逻辑寄存器 rm。

[119] 例如:

sub r1，r2

leanej L_Loop_i(rn)，rm

[120] 上述 leanej 指令表示有符号比较 r2 和 r1，如果 r2 不等于 r1，则将 L_Loop_i 所指向的指令地址存入 rn，否则将当前 leanej 指令后面的指令地址存入 rn，同时跳转到 rm。

[121] leaej 指令的格式为 leaej L_Loop_i(rn)，rm，用于表示等于，则将 L_Loop_i 所指向的指令地址存入逻辑寄存器 rn，否则将当前 leaej 指令后面的指令地址存入逻辑寄存器 rn，同时跳转到逻辑寄存器 rm。

[122] 例如:

sub r1，r2

leaej L_Loop_i(rn)，rm

[123] 上述 leaej 指令表示有符号比较 r2 和 r1，如果 r2 等于 r1，则将 L_Loop_i 所指向的指令地址存入 rn，否则将当前 leaej 指令后面的指令地址存入 rn，同时跳转到 rm。

[124] leaj 指令的格式为 leaj L_Loop_i(rn)，rm，用于表示无条件将 L_Loop_i 所指向的指令地址存入逻辑寄存器 rn，同时跳转到逻辑寄存器 rm。

[125] 例如:

sub r1，r2

leaj L_Loop_i(rn)，rm

[126] 上述 leaj 指令表示无条件将 L_Loop_i 所指向的指令地址存入 rn，同时跳转到 rm。

[127] leancj 指令的格式为 leancj L_Loop_i(rn)，rm，用于表示不进位，则将 L_Loop_i 所指向的指令地址存入逻辑寄存器 rn，否则将当前 leancj 指令后面的指令地址存入逻辑寄存器 rn，同时跳转到逻辑寄存器 rm。

[128] 例如:

sub r1，r2

leancj L_Loop_i(rn)，rm

[129] 上述 leancj 指令表示比较 r2 和 r1，如果不产生进位，则将 L_Loop_i 所指向的指令地址存入 rn，否则将当前 leancj 指令后面的指令地址存入 rn，同时跳转到 rm。

[130] leacj 指令的格式为 leacj L_Loop_i(rn)，rm，用于表示进位，则将 L_Loop_i 所指向的指令地址存入逻辑寄存器 rn，否则将当前 leacj 指令后面的指令地址存入逻辑寄存器

rn，同时跳转到逻辑寄存器 rm。

[131] 例如:

sub r1，r2

leacj L_Loop_i(rn)，rm

[132] 上述 leacj 指令表示比较 r2 和 r1，如果不产生进位，则将 L_Loop_i 所指向的指令地址存入 rn，否则将当前 leacj 指令后面的指令地址存入 rn，同时跳转到 rm。

[133] 相应地，为了保证所增加的 leaxxj 指令的指令功能能够顺利执行，需要对 CPU 的架构进行相应修改。例如，使得 CPU 能够识别 leaxxj 指令以及完成 leaxxj 指令的各种操作任务。

[134] 此外，针对各个线程中的各个指令块，转移指令包括单条 leaxxj 指令。例如，针对线程 Thread1 的第一指令块，转移指令包括指令“1.8 leaaj L_loop_1(r4)，r6”。针对线程 Thread1 的第二指令块，转移指令包括指令“1.16 leaaj L_loop_2(r4)，r6”。针对线程 Thread2 的第三指令块，转移指令包括指令“2.7 leaaj L_loop_3(r6)，r10”。针对线程 Thread3 的第四指令块，转移指令包括指令“3.6 leaaj L_loop_4(r10)，r4”。

[135] 按照图 7 中示出的线程融合方案，可以将多个线程的机器指令代码融合为单个线程的机器指令代码，由此在进行指令执行时，仅仅需要准备与单个线程对应的硬件运行环境，从而能够在无需增加硬件成本的情况下实现多线程处理。

[136] 此外，在图 7 中示出的示例中，可以利用单条指令来实现转移指令，从而大大减少融合后的线程的指令数目，由此提高融合后的线程的效率。例如，在图 7 的示例中，对于每个指令跳转块，仅仅需要 1 条 leaaj 指令，并且该 leaaj 指令是通过对原有的跳转指令修改而得到的。而在图 6 的示例中，对于每个指令跳转块，需要使用 4 条指令来实现跳转，即，2 条 lea 指令、1 条 cmovbe 指令和 1 条 jmp 指令。

[137] 要说明的是，在图 7 中示出的线程融合方案中，由于在跳转指令之前，处理内核知道跳转目的寄存器，从而无需进行寄存器预测处理。

[138] 回到图 5，在如上完成多线程融合后，在块 530，经由处理内核 430 来以指令块为单位获取所述融合线程中的机器指令来运行。

[139] 具体地，当前 CPU 每次取指令都是以指令块为单位来从指令解码队列中获取（例如，指令块所包含的最大指令条数可以为 5 或者 6 条，所获取的指令块的最大指令数目

可以由 CPU 的处理能力来确定，例如，可以由 CPU 在单个指令周期中能够执行的指令数目来确定）。如果按照规定的指令块大小，CPU 本次取到的指令块包含转移指令，例如，cmovbe 指令或 leaxxj 指令，并且该转移指令不是所获取的指令块的结束指令。在这种情况下，由于转移指令的后续指令是否执行仍然不明确，如果取到转移指令之后的指令，则可能会存在转移指令之后的指令没有执行，由此导致本次取值操作失败，需要回滚到本次取指操作的初始状态。针对这种情况，可以将本次取指操作仅仅取指到转移指令（例如，leaxxj 指令）为止，即，将本次取值的指令块调整为将转移指令作为结束指令，由此避免出现前述回滚情形。

[140] 此外，要说明的是，在进行机器指令运行时，如果处理内核 430 已经取回 leaxxj 指令，则将与该 leaxxj 指令中的逻辑寄存器 rn 中存放的地址对应的指令块从缓存或内存中预取到处理内核 430 的指令解码队列中。比如在取到指令 leaxxj L_loop_2(r4)，r6 的时候，就已经知道下次需要跳转到 r6 所指向的目的地址，因此开始预取 r6 所指向地址的指令块（Thread-2 的指令块）到该 CPU 内核的指令解码队列中。这样，在跳转过程中就会有指令连续进入指令解码队列，中间的运行持续展开无断开的现象。

[141] 图 8 示出了根据本说明书的实施例的融合线程运行过程的流程图。

[142] 如图 8 所示，在块 810，处理内核 430 以指令块为单位获取融合线程中的机器指令。例如，处理内核 430 获取图 7 中示出的融合线程中的线程 Thread1 中的指令块 1。假设指令块的大小为 5 条指令，则该指令块 1 由指令“1.1 lea r6，L_Thread2(rip)”、“1.2 lea r10，L_Thread3(rip)”“1.3 mov r1，r3”、“L_loop_1:1.4 add (addr1)，r1”和“1.5 mul r1，r2”。

[143] 然后，在块 820，处理内核 430 依次运行所取回的指令块中的指令。例如，处理内核 430 依次运行指令块 1 中的指令“1.1 lea r6，L_Thread2(rip)”、“1.2 lea r10，L_Thread3(rip)”“1.3 mov r1，r3”、“L_loop_1:1.4 add (addr1)，r1”和“1.5 mul r1，r2”。

[144] 接着，在块 830，判断当前运行的指令是否是转移指令。如果是转移指令，则判断该转移指令是否指示线程退出。

[145] 例如，在使用转移指令 leaxxj 来进行线程融合的情况下，如果 leaxxj 指令中的 L_loop_i 设置为指定数值（例如，0xff ff ff ff ff ff ff ff），则表示线程准备结束。此外，逻辑寄存器具有用于指示该逻辑寄存器的使用状态的标志位，所述使用状态包括用于表

示正常使用的第一状态和用于表示该逻辑寄存器指向另一逻辑寄存器的第二状态。例如，标志位可以使用“0”来表示第一状态，以及使用“1”来表示第二状态。

[146] 在进行机器指令运行时，如果处理内核识别出 leaxxj 指令中的 L_loop_i 为指定数值（例如，0xff ff ff ff ff ff ff ff），则在块 850，执行线程退出处理。具体地，处理内核释放对应的逻辑寄存器所指向的物理寄存器，使得对应的逻辑寄存器指向下一线程所使用的逻辑寄存器，并且将该对应的逻辑寄存器的标志位赋值为“1”。通常，指令数值被设置为 CPU 读取后不能识别其含义或者被识别为无效的数值，比如，0xff ff ff ff ff ff ff ff。关于线程退出机制，将在下面参照图 9 进行说明。

[147] 如果处理内核识别出 leaxxj 指令中的 L_loop_i 不是指定数值，则返回到块 810，取回下一指令块，执行下一循环操作。例如，取回指令块 2，该指令块包括指令“1.6 mov r2，addr2”、“1.7 sub $1，r3”和“1.8 leaaj L_loop_1(r4)，r6”，执行下一循环操作。

[148] 图 9 示出了根据本说明书的实施例的线程退出处理的线程示例示意图。在图 9 中，进行线程融合的线程包括线程 Thread1、Thread2、Thread3、Thread4 和 Thread5。

[149] 假设在融合线程程序的起始处，已经初始化了逻辑寄存器 r1 指向线程 Thread-1 的程序相应地址，逻辑寄存器 r2 指向线程 Thread-2 的程序相应地址，逻辑寄存器 r3 指向线程 Thread-3 的程序相应地址，逻辑寄存器 r4 指向线程 Thread-4 的程序相应地址，以及逻辑寄存器 r5 指向线程 Thread-5 的程序相应地址。

[150] 此外，在 CPU 中还设置有中转寄存器，该中转寄存器用来保留上一次正常跳转时的目的逻辑寄存器，例如当 Thread-1 正常执行指令 leaxxj .L_target_1(r1)，r2，此时在跳转的同时，需要将逻辑寄存器 r2 存入中转寄存器，若 Thread-2 也正常执行指令 leaxxj .L_target_1(r2)，r3，则在跳转的同时，需要将逻辑寄存器 r3 存入中转寄存器来替换上次存入的逻辑寄存器 r2。这里，中转寄存器的默认值为“0”。

[151] 在本说明书实施例中，在线程程序中使用的寄存器 r1，r2，r3，r4，r5 都是逻辑寄存器，逻辑寄存器指向的是物理寄存器，由此可以通过改变逻辑-物理寄存器之间的映射关系来改变寄存器中存储的内容，同时结合中转寄存器来实现线程结束。

[152] 另外，在每个逻辑寄存器中增加一个标志位 L，用来指示该逻辑寄存器的使用状态。所述使用状态可以包括用于指示该逻辑寄存器被正常使用的第一状态，以及用于指示该逻辑寄存器指向的内容为另外一个逻辑寄存器的第二状态。在本说明书实施例中，例如，可以使用值“0”来指示第一状态，以及使用值“1”来指示第二状态。

[153] 下面来结合图 9 详细说明根据本说明书的实施例的线程退出机制。

[154] 在正常运行一段时间之后，当线程 Thread-1 正常跳转后，中转寄存器存储着逻辑寄存器 r2，然后跳转到线程 Thread-2，开始执行线程 Thread-2。

[155] 如果在运行过程中，线程 Thread-2 将会结束，则会使用指令 leaxxj 0xff ff ff ff ff ff ff (r2)，r3 来表示。由于任何一个偏移地址都不可能是 0xff ff ff ff ff ff ff ff，因此 CPU 运行这条指令时，如果发现需要将 0xff ff ff ff ff ff ff ff 放入逻辑寄存器 r2 中，则由于中转寄存器存储的是逻辑寄存器 r2，CPU 会将逻辑寄存器 r2 原来指向的物理寄存器释放（如果有），同时使逻辑寄存器 r2 指向物理寄存器的空间中放入线程 Thread-3 所使用的逻辑寄存器（即，逻辑寄存器 r3），同时将逻辑寄存器 r2 的标志位 L 设定为 1。在指令重命名过程中，如果发现逻辑寄存器对应的 L 标志位是 1，则 CPU 会根据逻辑寄存器 r2 此时存储的逻辑寄存器（r3）来找到真正的物理寄存器。由于此时逻辑寄存器 r2，r3 都指向同一物理寄存器，因此当线程 Thread-1 再次运行 leaxxj 指令需要跳到逻辑寄存器 r2 的地址时，会直接跳入线程 Thread-3 对应的上下文中。

[156] 在运行一段时间后，如果线程 Thread-3 也要执行指令 leaxxj 0xff ff ff ff ff ff ff (r3)，r4，此时由于线程 Thread-2 已经退出，会从线程 Thread-1 跳到线程 Thread-3，因此中转寄存器存入的是逻辑寄存器 r2。由此，CPU 将逻辑寄存器 r2 指向物理寄存器的空间存入线程 Thread-4 所使用的逻辑寄存器 r4，并且逻辑寄存器 r2 的标志位 L 被设定为 1。这样，当线程 Thread-1 再次执行指令 leaxxj .L_target_1(r1)，r2 时，会直接从线程 Thread-1 跳入到线程 Thread-4，以此类推。

[157] 如果线程 4 退出时，中转寄存器存入的是逻辑寄存器 r2，那么使 r2 指向物理寄存器的空间中放入线程 Thread-5 所使用的逻辑寄存器（即，逻辑寄存器 r5），那么 Thread-1 会直接跳入 Thread-5,但是当 Thread-5 需要退出的时候，此时中转寄存器存入的是逻辑寄存器 r2，会将 r2 指向物理寄存器的空间中放入线程 Thread-1 所使用的逻辑寄存器（即，逻辑寄存器 r1），因此当 Thread-1 退出的时候发现逻辑寄存器 r2 指向的逻辑寄存器为自己（即，r1），那么正常退出，完成整个操作。

[158] 图 10 示出了根据本说明书的实施例的多线程编译装置 1000 的方框图。如图 10 所示，多线程编译装置 1000 包括线程编译单元 1010 和线程融合单元 1020。

[159] 线程编译单元 1010 被配置为针对待处理的第一数目个线程中的各个线程，使用不同的第一逻辑寄存器集合来进行编译以得到对应的机器指令代码。线程编译单元 1010

的操作可以参考上面参照图 5 描述的块 510 的操作。

[160] 线程融合单元 1020 被配置为使用处理内核所支持的指令集中的第一指令来对各个线程的机器指令代码进行融合处理，以得到单个融合线程的机器代码指令。这里，所述第一指令包括取有效地址指令和转移指令，所述取有效地址指令和转移指令利用与所述第一逻辑寄存器集合不同的第二逻辑寄存器来编译得到，所述转移指令被设置在所述各个线程所对应的机器指令代码中，用于实现所述各个线程的机器指令代码之间的跳转操作。线程融合单元 1020 的操作可以参考上面参照图 5 描述的块 520 的操作以及参照图 6 到图 9 描述的操作。

[161] 此外，多线程编译装置 1000 还可以包括转移指令获得单元（未示出）。转移指令获得单元通过对所述各个线程的机器指令代码中的全部或部分跳转指令进行修改而获得第一转移指令。此外，在本说明书实施例的另一示例中，所述转移指令获得单元进一步基于转移指令增加规则增加第二转移指令。所述转移指令增加规则包括：使得转移指令彼此间隔不超过第二预定数目条指令；和/或使用指令编译器调优策略来在合适的机器指令之后增加所述第二转移指令。

[162] 如上参照图 1 到图 10，对根据本说明书实施例的用于多线程处理的方法、装置和多线程处理系统的实施例进行了描述。上面的多线程编译装置/多线程处理系统可以采用硬件实现，也可以采用软件或者硬件和软件的组合来实现。

[163] 图 11 为根据本说明书的实施例的用于多线程处理的计算设备的结构框图。

[164] 如图 11 所示，计算设备 1100 可以包括至少一个处理器 1110、存储器（例如，非易失性存储器）1120、内存 1130、通信接口 1140 以及内部总线 1160，并且至少一个处理器 1110、存储器 1120、内存 1130 和通信接口 1140 经由总线 1160 连接在一起。该至少一个处理器 1110 执行在计算机可读存储介质中存储或编码的至少一个计算机可读指令（即，上述以软件形式实现的元素）。

[165] 在一个实施例中，在存储器中存储有计算机可执行指令，其当执行时使得至少一个处理器 1110：针对待处理的第一数目个线程中的各个线程，使用不同的第一逻辑寄存器集合来进行编译以得到对应的机器指令代码；以及使用处理内核所支持的指令集中的第一指令来对各个线程的机器指令代码进行融合处理，以得到单个融合线程的机器指令代码，其中，第一指令包括取有效地址指令和转移指令，所述取有效地址指令和转移指令利用与第一逻辑寄存器集合不同的第二逻辑寄存器来编译得到，所述转移指令被设置

在各个线程所对应的机器指令代码中，用于实现各个线程的机器指令代码之间的跳转操作。

[166] 应该理解的是，在存储器中存储的计算机可执行指令当执行时使得至少一个处理器 1110 进行本说明书的各个实施例中以上结合图 1-10 描述的各种操作和功能。

[167] 根据一个实施例，提供了一种例如非暂时性机器可读介质的程序产品。非暂时性机器可读介质可以具有指令（即，上述以软件形式实现的元素），该指令当被机器执行时，使得机器执行本说明书的各个实施例中以上结合图 1-10 描述的各种操作和功能。

[168] 具体地，可以提供配有可读存储介质的系统或者装置，在该可读存储介质上存储着实现上述实施例中任一实施例的功能的软件程序代码，且使该系统或者装置的计算机或处理器读出并执行存储在该可读存储介质中的指令。

[169] 在这种情况下，从可读介质读取的程序代码本身可实现上述实施例中任何一项实施例的功能，因此机器可读代码和存储机器可读代码的可读存储介质构成了本发明的一部分。

[170] 可读存储介质的实施例包括软盘、硬盘、磁光盘、光盘（如 CD-ROM、CD-R、CD-RW、DVD-ROM、DVD-RAM、DVD-RW、DVD-RW）、磁带、非易失性存储卡和 ROM。可选择地，可以由通信网络从服务器计算机上或云上下载程序代码。

[171] 本领域技术人员应当理解，上面公开的各个实施例可以在不偏离发明实质的情况下做出各种变形和修改。因此，本发明的保护范围应当由所附的权利要求书来限定。

[172] 需要说明的是，上述各流程和各系统结构图中不是所有的步骤和单元都是必须的，可以根据实际的需要忽略某些步骤或单元。各步骤的执行顺序不是固定的，可以根据需要进行确定。上述各实施例中描述的装置结构可以是物理结构，也可以是逻辑结构，即，有些单元可能由同一物理实体实现，或者，有些单元可能分由多个物理实体实现，或者，可以由多个独立设备中的某些部件共同实现。

[173] 以上各实施例中，硬件单元或模块可以通过机械方式或电气方式实现。例如，一个硬件单元、模块或处理器可以包括永久性专用的电路或逻辑（如专门的处理器，FPGA 或 ASIC）来完成相应操作。硬件单元或处理器还可以包括可编程逻辑或电路（如通用处理器或其它可编程处理器），可以由软件进行临时的设置以完成相应操作。具体的实现方式（机械方式、或专用的永久性电路、或者临时设置的电路）可以基于成本和时间上的考虑来确定。

[174] 上面结合附图阐述的具体实施方式描述了示例性实施例，但并不表示可以实现的或者落入权利要求书的保护范围的所有实施例。在整个本说明书中使用的术语“示例性”意味着“用作示例、实例或例示”，并不意味着比其它实施例“优选”或“具有优势”。出于提供对所描述技术的理解的目的，具体实施方式包括具体细节。然而，可以在没有这些具体细节的情况下实施这些技术。在一些实例中，为了避免对所描述的实施例的概念造成难以理解，公知的结构和装置以框图形式示出。

[175] 本公开内容的上述描述被提供来使得本领域任何普通技术人员能够实现或者使用本公开内容。对于本领域普通技术人员来说，对本公开内容进行的各种修改是显而易见的，并且，也可以在不脱离本公开内容的保护范围的情况下，将本文所定义的一般性原理应用于其它变型。因此，本公开内容并不限于本文所描述的示例和设计，而是与符合本文公开的原理和新颖性特征的最广范围相一致。

权利要求书

1、一种用于多线程处理的方法，包括:

针对待处理的第一数目个线程中的各个线程，使用不同的第一逻辑寄存器集合来进行编译以得到对应的机器指令代码；以及

使用处理内核所支持的指令集中的第一指令来对各个线程的机器指令代码进行融合处理，以得到单个融合线程的机器指令代码，

其中，所述第一指令包括取有效地址指令和转移指令，所述取有效地址指令和转移指令利用与所述第一逻辑寄存器集合不同的第二逻辑寄存器来编译得到，所述转移指令被设置在所述各个线程所对应的机器指令代码中，用于实现所述各个线程的机器指令代码之间的跳转操作。

2、如权利要求 1 所述的方法，其中，所述第一数目不超过第一预定阈值，所述第一预定阈值是基于所述处理内核可使用的逻辑寄存器的数目来确定的。

3、如权利要求 1 所述的方法，其中，所述转移指令包括通过对所述各个线程的机器指令代码中的全部或部分跳转指令进行修改而获得的第一转移指令。

4、如权利要求 3 所述的方法，其中，所述转移指令还包括基于转移指令增加规则增加的第二转移指令。

5、如权利要求 4 所述的方法，其中，所述转移指令增加规则包括:

使得所述转移指令彼此间隔不超过第二预定数目条指令；和/或

使用指令编译器调优策略来在合适的机器指令之后增加所述第二转移指令。

6、如权利要求 5 所述的方法，其中，所述第二预定数目是基于所述处理内核在单个指令周期所能执行的指令数目确定的。

7、如权利要求 1 所述的方法，其中，经过融合后的每个线程具有基于转移指令分割出的至少一个指令跳转块。

8、如权利要求 7 所述的方法，其中，针对除所述第一数目个线程中的第一线程之外的每个线程，所述取有效地址指令包括单条第一取有效地址指令，所有第一取有效地址指令都被放置在所述融合线程的机器指令代码的开始处，用于将该线程的第一条指令的地址传送到对应的第二逻辑寄存器，以及

针对所述第一数目个线程中的每个线程中的每个指令跳转块，所述取有效地址指令包括两条第二取有效地址指令，其中，一条第二取有效地址指令用于将该指令跳转块的第一条指令的地址传送到对应的第二逻辑寄存器，另一第二取有效地址指令用于将位于该指令跳转块之后的第一条指令的地址传送到对应的第二逻辑寄存器，所述两条第二取

有效地址指令被放置在各自线程的对应指令跳转块和上一指令跳转块之间，以及所述转移指令包括条件传送指令和无条件跳转指令。

9、如权利要求 7 所述的方法，其中，所述指令集包括 leaxxj 指令，针对除所述第一数目个线程中的第一线程之外的每个线程，所述取有效地址指令包括单条第一取有效地址指令，所述第一取有效地址指令用于将该线程的第一条指令的地址传送到对应的第二逻辑寄存器，并且所有第一取有效地址指令都被放置在所述融合线程的机器指令代码的开始处，以及针对所述第一数目个线程中的各个线程中的每个指令跳转块，所述转移指令包括单条 leaxxj 指令，

其中，所述 leaxxj 指令的指令格式为 leaxxj L_loop_i（rm），rn，其中，逻辑寄存器 rm 表示对应的指令跳转块所属线程的第一取有效地址指令所使用的第二逻辑寄存器，以及逻辑寄存器 rn 表示需要跳转到的线程的第一取有效地址指令所使用的第二逻辑寄存器，所述 leaxxj 指令用于指示在条件满足时将 L_loop_i 所指向的指令地址放入逻辑寄存器 rm，在条件不满足时将所述 leaxxj 指令之后的第一条指令的指令地址放入逻辑寄存器 rm，并且跳转到逻辑寄存器 rn 中存储的地址所对应的指令。

10、如权利要求 9 所述的方法，其中，所述第二逻辑寄存器具有用于指示该第二逻辑寄存器的使用状态的标志位，所述使用状态包括用于表示正常使用的第一状态和用于表示该逻辑寄存器指向另一逻辑寄存器的第二状态，

在进行机器指令运行时，如果所述处理内核识别出 leaxxj 指令中的 L_loop_i 被赋值为指定数值，则释放第二逻辑寄存器 rn 所指向的物理寄存器，使得第二逻辑寄存器 rn 指向目标跳转线程所使用的第二逻辑寄存器，并且将第二逻辑寄存器 rn 的标志位赋值为指示第二状态。

11、如权利要求 9 所述的方法，其中，在进行机器指令运行时，如果所述处理内核已经取回 leaxxj 指令，则将与所述 leaxxj 指令中的逻辑寄存器 rn 所存放的地址对应的指令块预取到所述处理内核的指令解码队列中。

12、如权利要求 7 到 11 中任一所述的方法，还包括:

经由所述处理内核来以指令块为单位获取所述融合线程中的机器指令来运行。

13、如权利要求 12 所述的方法，其中，所述指令块的大小由所述处理内核在单个指令周期读取指令的能力来确定。

14、如权利要求 13 所述的方法，其中，在所获取的指令块中包括转移指令并且该指令块的结束指令不是所述转移指令时，将该指令块调整为以所述转移指令为结束指令。

15、一种用于多线程处理的装置，包括:

线程编译单元，针对待处理的第一数目个线程中的各个线程，使用不同的第一逻辑寄存器集合来进行编译以得到对应的机器指令代码；以及

线程融合单元，使用处理内核所支持的指令集中的第一指令来对各个线程的机器指令代码进行融合处理，以得到单个融合线程的机器指令代码，

其中，所述第一指令包括取有效地址指令和转移指令，所述取有效地址指令和转移指令利用与所述第一逻辑寄存器集合不同的第二逻辑寄存器来编译得到，所述转移指令被设置在所述各个线程所对应的机器指令代码中，用于实现所述各个线程的机器指令代码之间的跳转操作。

16、如权利要求 15 所述的装置，还包括:

转移指令获得单元，通过对所述各个线程的机器指令代码中的全部或部分跳转指令进行修改而获得的第一转移指令。

17、如权利要求 16 所述的装置，其中，所述转移指令获得单元进一步基于转移指令增加规则增加第二转移指令。

18、如权利要求 16 所述的装置，其中，所述转移指令增加规则包括:

使得所述转移指令彼此间隔不超过第二预定数目条指令；和/或

使用指令编译器调优策略来在合适的机器指令之后增加所述第二转移指令。

19、如权利要求 15 所述的装置，其中，经过融合后的每个线程具有基于转移指令分割出的至少一个指令跳转块。

20、如权利要求 19 所述的装置，其中，针对除所述第一数目个线程中的第一线程之外的每个线程，所述取有效地址指令包括单条第一取有效地址指令，所有第一取有效地址指令都被放置在所述融合线程的机器指令代码的开始处，用于将该线程的第一条指令的地址传送到对应的第二逻辑寄存器，以及

针对所述第一数目个线程中的每个线程中的每个指令跳转块，所述取有效地址指令包括两条第二取有效地址指令，其中，一条第二取有效地址指令用于将该指令跳转块的第一条指令的地址传送到对应的第二逻辑寄存器，另一第二取有效地址指令用于将位于该指令跳转块之后的第一条指令的地址传送到对应的第二逻辑寄存器，所述两条第二取有效地址指令被放置在各自线程的对应指令跳转块和上一指令跳转块之间，以及所述转移指令包括条件传送指令和无条件跳转指令。

21、如权利要求 19 所述的装置，其中，所述指令集包括 leaxxj 指令，针对除所述第一数目个线程中的第一线程之外的每个线程，所述取有效地址指令包括单条第一取有效地址指令，所述第一取有效地址指令用于将该线程的第一条指令的地址传送到对应的

第二逻辑寄存器，并且所有第一取有效地址指令都被放置在所述融合线程的机器指令代码的开始处，以及针对所述第一数目个线程中的各个线程中的每个指令跳转块，所述转移指令包括单条 leaxxj 指令，

其中，所述 leaxxj 指令的指令格式为 leaxxj L_loop_i（rm），rn，其中，逻辑寄存器 rm 表示对应的指令跳转块所属线程的第一取有效地址指令所使用的第二逻辑寄存器，以及逻辑寄存器 rn 表示需要跳转到的线程的第一取有效地址指令所使用的第二逻辑寄存器，所述 leaxxj 指令用于指示在条件满足时将 L_loop_i 所指向的指令地址放入逻辑寄存器 rm，在条件不满足时将所述 leaxxj 指令之后的第一条指令的指令地址放入逻辑寄存器 rm，并且跳转到逻辑寄存器 rn 中存储的地址所对应的指令。

22、一种用于多线程处理的系统，包括:

编译器，包括如权利要求 15 到 21 中任一所述的装置，将第一数目个线程编译为单个融合线程的机器指令代码;

指令集；以及

处理内核，以指令块为单位获取所述融合线程中的机器指令来运行。

23、如权利要求 22 所述的系统，其中，所述指令集包括 leaxxj 指令，所述 leaxxj 指令的指令格式为 leaxxj L_loop_i（rm），rn，其中，所述 leaxxj 指令的指令格式为 leaxxj L_loop_i（rm），rn，其中，逻辑寄存器 rm 表示对应的指令跳转块所属线程的第一取有效地址指令所使用的第二逻辑寄存器，以及逻辑寄存器 rn 表示需要跳转到的线程的第一取有效地址指令所使用的第二逻辑寄存器，所述 leaxxj 指令用于指示在条件满足时将 L_loop_i 所指向的指令地址放入逻辑寄存器 rm，在条件不满足时将所述 leaxxj 指令之后的第一条指令的指令地址放入逻辑寄存器 rm，并且跳转到逻辑寄存器 rn 中存储的地址所对应的指令。

24、如权利要求 23 所述的系统，其中，所述第二逻辑寄存器具有用于指示该第二逻辑寄存器的使用状态的标志位，所述使用状态包括用于表示正常使用的第一状态和用于表示该逻辑寄存器指向另一逻辑寄存器的第二状态，

在进行机器指令运行时，如果所述处理内核识别出 leaxxj 指令中的 L_loop_i 被赋值为指定数值，则所述处理内核释放第二逻辑寄存器 rn 所指向的物理寄存器，使得第二逻辑寄存器 rn 指向目标跳转线程所使用的第二逻辑寄存器，并且将第二逻辑寄存器 rn 的标志位赋值为指示第二状态。

25、如权利要求 23 所述的系统，其中，在进行机器指令运行时，如果所述处理内核已经取回 leaxxj 指令，则所述处理内核将与所述 leaxxj 指令中的逻辑寄存器 rn 所存放

的地址对应的指令块预取到所述处理内核的指令解码队列中。

26、一种计算设备，包括:

一个或多个处理器，以及

与所述一个或多个处理器耦合的存储器，所述存储器存储指令，当所述指令被所述一个或多个处理器执行时，使得所述一个或多个处理器执行如权利要求 1 到 14 中任一项所述的方法。

27、一种机器可读存储介质，其存储有可执行指令，所述指令当被执行时使得所述机器执行如权利要求 1 到 14 中任一项所述的方法。

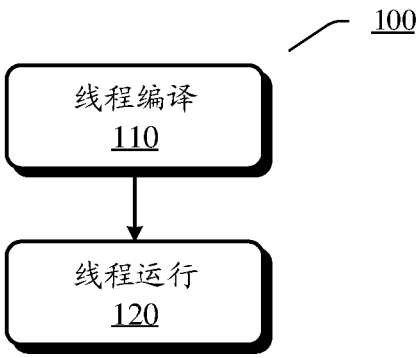


图 1

```
Thread1
    L_start_1
    1.1 mov r1, r3
    L_loop_1
    1.2 add (addr1), r1
    1.3 mul r1, r2
    1.4 mov r2, addr1
    1.5 sub $1, r3
    1.6 ja L_loop_1
    L_sub_again
    1.7 sub r1, r3
    1.8 cmp 0, r3
    1.9 ja L_sub_again
    L_loop_2
    1.10 sub (addr1), r1
    1.11 div r1, r2
    1.12 add r2, addr1
    1.13 add $3, r3
    1.14 jbe L_loop_2
     ... ...
Thread2
    L_start_2
    2.1 mov $9, r1
    2.2 mov $100, r3
    L_loop_3
    2.3 add  (addr1), r1
    2.4 mul r1, r3
    2.5 add r3, addr1
    2.6 sub $1, r3
    2.7 ja L_loop_3
    2.8 add $9,r3
     ... ...
Thread3
    L_start_3
    3.1 mov r2, r4
    L_loop_4
    3.2 sub (addr1), r1
    3.3 div r2, r1
    3.4 sub r1, addr1
    3.5 sub $1, r4
    3.6 ja L_loop_4
     ... ...
```

图 2

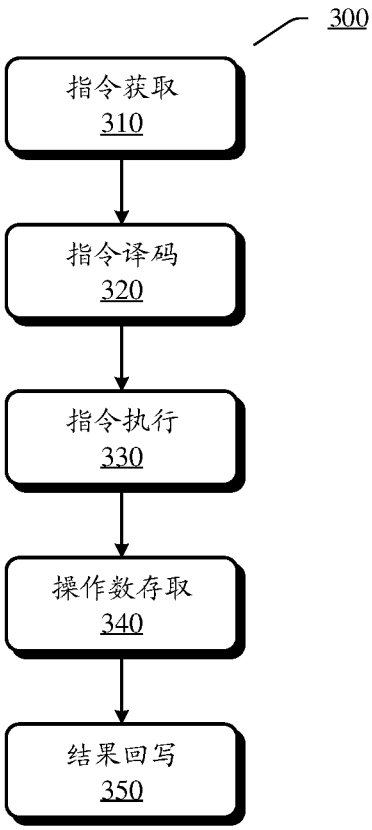

300
指令获取
310
指令译码
320
指令执行
330
操作数存取
340
结果回写
350

图 3

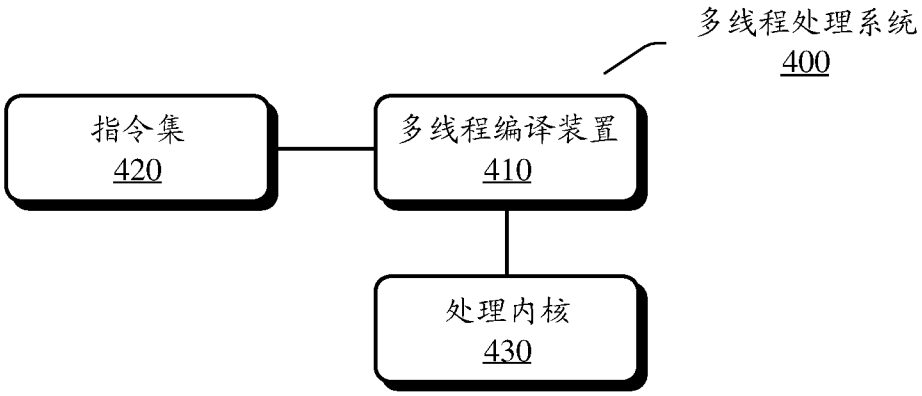

多线程处理系统
400
指令集
420
多线程编译装置
410
处理内核
430

图 4

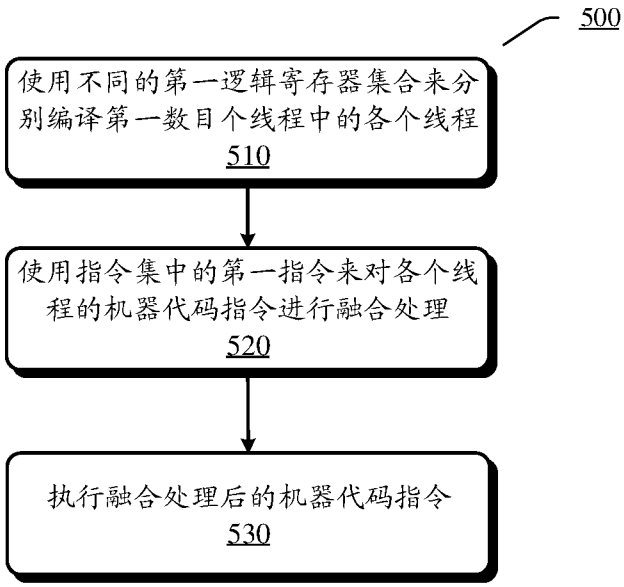

500
使用不同的第一逻辑寄存器集合来分别编译第一数目个线程中的各个线程
510
使用指令集中的第一指令来对各个线程的机器代码指令进行融合处理
520
执行融合处理后的机器代码指令
530

图 5

Thread Fusion

```
1.1 lea r6, L_Thread2(rip)
1.2 lea r10, L_Thread3(rip)
L_Thread_1
1.3 mov r1, r3
1.4 lea r5, Leave_loop_1(rip)
1.5 lea r4, L_loop_1(rip)
L_loop_1
1.6 add (addr1), r1
1.7 mul r1, r2
1.8 mov r2, addr1
1.9 sub $1, r3
1.10 cmovbe r5, r4
1.11 jmp r6
Leave_loop_1:
L_sub_again
1.12 sub r1, r3
1.13 cmp $0, r3
1.14 ja L_sub_again
1.15 lea r5, Leave_loop_2(rip)
1.16 lea r4, L_loop_2(rip)
L_loop_2
1.17 sub (addr1), r1
1.18 div r1, r2
1.19 add r2, addr1
1.20 add $3, r3
1.21 cmovbe r5, r4
1.22 jmp r6
Leave_loop_2:
……
L_Thread_2
2.1 lea r7, Leave_loop_3(rip)
2.2 lea r6, L_loop_3(rip)
2.3 mov $9, r8
2.4 mov $100, r9
L_loop_3
2.5 add  (addr2), r8
2.6 mul r8, r9
2.7 add r9, addr2
2.8 sub $1, r9
2.9 cmovbe r7, r6
2.10 jmp r10
Leave_loop_3:
2.11 add $9,r9
……
L_Thread_3
3.1 lea r11, Leave_loop_4(rip)
3.2 lea r10, L_loop_4(rip)
3.3 mov r12, r13
L_loop_4
3.4 sub (addr3), r14
3.5 div r12, r14
3.6 sub r14, addr3
3.7 sub $1, r13
3.8 cmovbe r11, r10
3.9 jmp r4
Leave_loop_4:
……
```

图 6

Thread Fusion

```
1.1 lea r6, L_Thread2(rip)
1.2 lea r10, L_Thread3(rip)
L_Thread_1
1.3 mov r1, r3
L_loop_1
1.4 add (addr1), r1
1.5 mul r1, r2
1.6 mov r2, addr1
1.7 sub $1, r3
1.8 leaaj L_loop_1(r4), r6
L_sub_again
1.9 sub r1, r3
1.10 cmp $0, r3
1.11 ja L_sub_again
L_loop_2
1.12 sub (addr1), r1
1.13 div r1, r2
1.14 add r2, addr1
1.15 add $3, r3
1.16 leaaj L_loop_2(r4), r6
... ...
L_Thread_2
2.1 mov $9, r8
2.2 mov $100, r9
L_loop_3
2.3 add  (addr2), r8
2.4 mul r8, r9
2.5 add r9, addr2
2.6 sub $1, r9
2.7 leaaj L_loop_3(r6), r10
2.8 add $9,r9
... ...
L_Thread_3
3.1 mov r11, r12
L_loop_4
3.2 sub (addr3), r13
3.3 div r11, r13
3.4 sub r13, addr3
3.5 sub $1, r12
3.6 leaaj L_loop_4(r10), r4
... ...
```

图 7

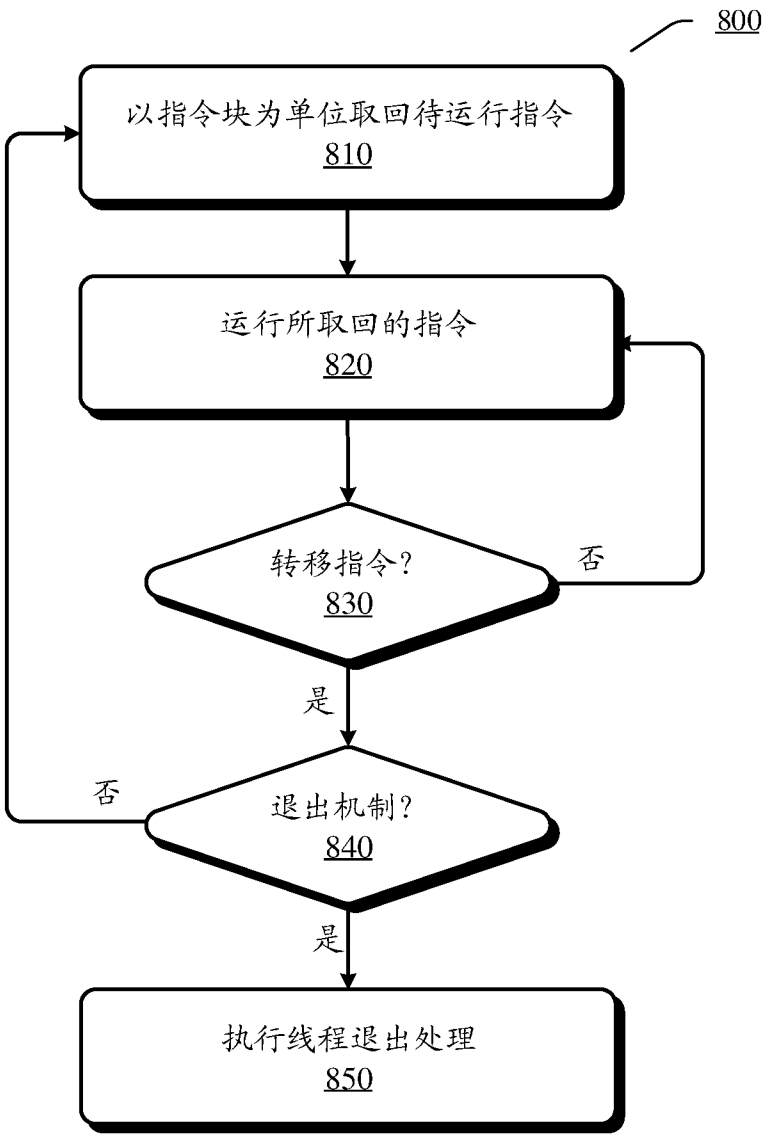

800
以指令块为单位取回待运行指令
810
运行所取回的指令
820
转移指令?
830
否
是
退出机制?
840
否
是
执行线程退出处理
850

图 8

Thread-1	Thread-2	Thread-3	Thread-4	Thread-5
……	……	……	……	……
Leaaj L_target_1(r1),r2	Leaaj L_target_2(r2),r3	Leaaj L_target_3(r3),r4	Leaaj L_target_4(r4),r5	Leaaj L_target_5(r5),r1
……	……	……	……	……
L_target_1:	L_target_2:	L_target_3:	L_target_4:	L_target_5:
……	……	……	……	……

图 9



图 10

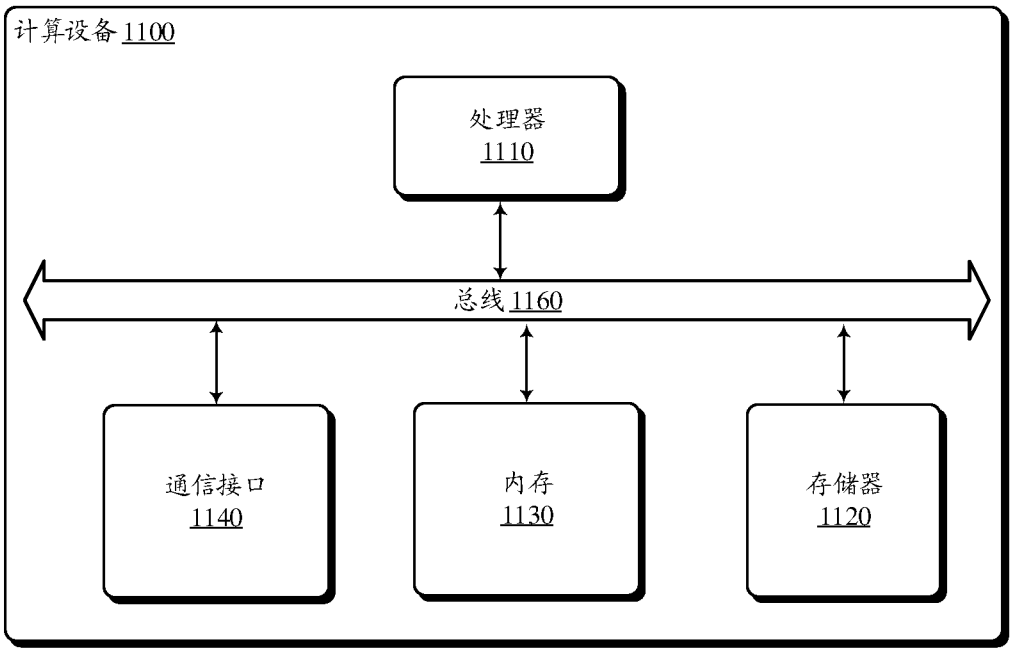


图 11

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2020/071423

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/30(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

CNTXT; CNABS; CNKI; WOTXT; USTXT; VEN; EPTXT: 多线程, 逻辑, 寄存器, 编译, 取有效地址, 转移, 指令, 跳转, multiple threads, multi-thread, logic, register, compile, load effective address, lea, transfer, instruction, jump

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
PX	CN 110569067 A (ALIBABA GROUP HOLDING LIMITED) 13 December 2019 (2019-12-13) entire document	1-27
A	CN 1853165 A (INTEL CORPORATION) 25 October 2006 (2006-10-25) description, page 5, paragraph 2 to page 11, paragraph 1, figures 1-7	1-27
A	CN 107368285 A (INTEL CORPORATION) 21 November 2017 (2017-11-21) description, paragraphs 101-123	1-27
A	US 2018004526 A1 (INTEL IP CORPORATION) 04 January 2018 (2018-01-04) entire document	1-27
A	CN 109614220 A (ALIBABA GROUP HOLDING LIMITED) 12 April 2019 (2019-04-12) entire document	1-27

☐ Further documents are listed in the continuation of Box C. ☑ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
09 May 2020	**19 May 2020**

Name and mailing address of the ISA/CN	Authorized officer
China National Intellectual Property Administration (ISA/CN) **No. 6, Xitucheng Road, Jimenqiao Haidian District, Beijing 100088** **China**	
Facsimile No. **(86-10)62019451**	Telephone No.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2020/071423

Patent document cited in search report	Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
CN 110569067 A	13 December 2019	None	
CN 1853165 A	25 October 2006	WO 2005033931 A3	22 December 2005
		WO 2005033931 A2	14 April 2005
		EP 1668502 A2	14 June 2006
		JP 4701174 B2	15 June 2011
		US 8612949 B2	17 December 2013
		US 2005071438 A1	31 March 2005
		US 2010281471 A1	04 November 2010
		JP 2007507048 A	22 March 2007
CN 107368285 A	21 November 2017	US 2010161948 A1	24 June 2010
		WO 2008061154 A9	03 July 2008
		EP 2523101 A1	14 November 2012
		US 8677105 B2	18 March 2014
		US 9501280 B2	22 November 2016
		EP 2122461 A4	24 March 2010
		US 9965281 B2	08 May 2018
		EP 2523101 B1	04 June 2014
		EP 2122461 A2	25 November 2009
		WO 2008061154 A3	04 September 2008
		CN 101627365 A	13 January 2010
		US 2018293073 A1	11 October 2018
		CN 101627365 B	29 March 2017
		WO 2008061154 A2	22 May 2008
		EP 2527972 A3	06 August 2014
		US 2017139714 A1	18 May 2017
		US 10585670 B2	10 March 2020
		EP 2527972 A2	28 November 2012
		US 2014181475 A1	26 June 2014
		IN 200901939 P2	19 June 2009
		IN 323136 B	25 October 2019
US 2018004526 A1	04 January 2018	None	
CN 109614220 A	12 April 2019	None	

国际检索报告

国际申请号
PCT/CN2020/071423

A. 主题的分类

G06F 9/30(2006.01)i

按照国际专利分类(IPC)或者同时按照国家分类和IPC两种分类

B. 检索领域

检索的最低限度文献(标明分类系统和分类号)

G06F

包含在检索领域中的除最低限度文献以外的检索文献

在国际检索时查阅的电子数据库(数据库的名称，和使用的检索词（如使用））

CNTXT;CNABS;CNKI;WOTXT;USTXT;VEN;EPTXT:多线程，逻辑，寄存器，编译，取有效地址， 转移，指令，跳转，multiple threads, multi-thread, logic, register, compile, load effective address, lea, transfer, instruction, jump

C. 相关文件

类 型*	引用文件，必要时，指明相关段落	相关的权利要求
PX	CN 110569067 A (阿里巴巴集团控股有限公司) 2019年 12月 13日 (2019-12-13) 全文	1-27
A	CN 1853165 A (英特尔公司) 2006年 10月 25日 (2006-10-25) 说明书第5页第2段-第11页第1段，图1-7	1-27
A	CN 107368285 A (英特尔公司) 2017年 11月 21日 (2017-11-21) 说明书第101-123段	1-27
A	US 2018004526 A1 (INTEL IP CORP) 2018年 1月 4日 (2018-01-04) 全文	1-27
A	CN 109614220 A (阿里巴巴集团控股有限公司) 2019年 4月 12日 (2019-04-12) 全文	1-27

☐ 其余文件在C栏的续页中列出。

☑ 见同族专利附件。

* 引用文件的具体类型：

“A” 认为不特别相关的表示了现有技术一般状态的文件

“E” 在国际申请日的当天或之后公布的在先申请或专利

“L” 可能对优先权要求构成怀疑的文件，或为确定另一篇引用文件的公布日而引用的或者因其他特殊理由而引用的文件（如具体说明的）

“O” 涉及口头公开、使用、展览或其他方式公开的文件

“P” 公布日先于国际申请日但迟于所要求的优先权日的文件

“T” 在申请日或优先权日之后公布，与申请不相抵触，但为了理解发明之理论或原理的在后文件

“X” 特别相关的文件，单独考虑该文件，认定要求保护的发明不是新颖的或不具有创造性

“Y” 特别相关的文件，当该文件与另一篇或者多篇该类文件结合并且这种结合对于本领域技术人员为显而易见时，要求保护的发明不具有创造性

“&” 同族专利的文件

国际检索实际完成的日期	国际检索报告邮寄日期
2020年 5月 9日	2020年 5月 19日

ISA/CN的名称和邮寄地址	受权官员
中国国家知识产权局(ISA/CN) 中国北京市海淀区蓟门桥西土城路6号 100088	刘细金
传真号 (86-10)62019451	电话号码 86-(20)-28950395

国际检索报告
关于同族专利的信息

国际申请号
PCT/CN2020/071423

检索报告引用的专利文件			公布日 (年/月/日)	同族专利			公布日 (年/月/日)
CN	110569067	A	2019年 12月 13日	无			
CN	1853165	A	2006年 10月 25日	WO	2005033931	A3	2005年 12月 22日
				WO	2005033931	A2	2005年 4月 14日
				EP	1668502	A2	2006年 6月 14日
				JP	4701174	B2	2011年 6月 15日
				US	8612949	B2	2013年 12月 17日
				US	2005071438	A1	2005年 3月 31日
				US	2010281471	A1	2010年 11月 4日
				JP	2007507048	A	2007年 3月 22日
CN	107368285	A	2017年 11月 21日	US	2010161948	A1	2010年 6月 24日
				WO	2008061154	A9	2008年 7月 3日
				EP	2523101	A1	2012年 11月 14日
				US	8677105	B2	2014年 3月 18日
				US	9501280	B2	2016年 11月 22日
				EP	2122461	A4	2010年 3月 24日
				US	9965281	B2	2018年 5月 8日
				EP	2523101	B1	2014年 6月 4日
				EP	2122461	A2	2009年 11月 25日
				WO	2008061154	A3	2008年 9月 4日
				CN	101627365	A	2010年 1月 13日
				US	2018293073	A1	2018年 10月 11日
				CN	101627365	B	2017年 3月 29日
				WO	2008061154	A2	2008年 5月 22日
				EP	2527972	A3	2014年 8月 6日
				US	2017139714	A1	2017年 5月 18日
				US	10585670	B2	2020年 3月 10日
				EP	2527972	A2	2012年 11月 28日
				US	2014181475	A1	2014年 6月 26日
				IN	200901939	P2	2009年 6月 19日
				IN	323136	B	2019年 10月 25日
US	2018004526	A1	2018年 1月 4日	无			
CN	109614220	A	2019年 4月 12日	无			