

(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  
2017年1月5日 (05.01.2017)



(10) 国际公布号  
WO 2017/000839 A1

- (51) 国际专利分类号:  
G06F 9/45 (2006.01)
- (21) 国际申请号: PCT/CN2016/087051
- (22) 国际申请日: 2016年6月24日 (24.06.2016)
- (25) 申请语言: 中文
- (26) 公布语言: 中文
- (30) 优先权:  
201510373054.7 2015年6月30日 (30.06.2015) CN
- (71) 申请人: 华为技术有限公司 (HUAWEI TECHNOLOGIES CO., LTD.) [CN/CN]; 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong 518129 (CN)。
- (72) 发明人: 陈建 (CHEN, Jian); 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong 518129 (CN)。周洪 (ZHOU, Hong); 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong 518129 (CN)。胡新宇 (HU, Xinyu); 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong 518129 (CN)。管红光 (GUAN, Hongguang); 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong

518129 (CN)。 张小俊 (ZHANG, Xiaojun); 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong 518129 (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, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, 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, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW。

(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)。

[见续页]

(54) Title: HARDWARE ACCELERATION METHOD, COMPILER AND DEVICE

(54) 发明名称: 一种硬件加速方法、编译器以及设备

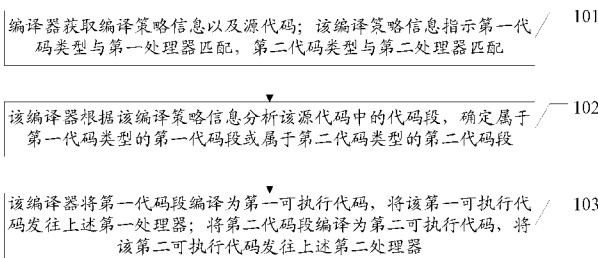


图 2

101 A compiler acquiring compiling policy information and a source code, wherein the compiling policy information indicates that a first code type matches a first processor, and a second code type matches a second processor

102 The compiler analyzing code segments in the source code according to the compiling policy information, so as to determine a first code segment belonging to the first code type or a second code segment belonging to the second code type

103 The compiler compiling the first code segment into a first executable code, and sending the first executable code to the first processor; and compiling the second code segment into a second executable code, and sending the second executable code to the second processor

(57) Abstract: Disclosed are a hardware acceleration method, compiler and device, which are used for improving the code execution efficiency and then realizing hardware acceleration. The method in embodiments of the present invention comprises: a compiler acquiring compiling policy information and a source code, wherein the compiling policy information indicates that a first code type matches a first processor, and a second code type matches a second processor; the compiler analyzing code segments in the source code according to the compiling policy information, so as to determine a first code segment belonging to the first code type or a second code segment belonging to the second code type; and the compiler compiling the first code segment into a first executable code, and sending the first executable code to the first processor; and compiling the second code segment into a second executable code, and sending the second executable code to the second processor.

(57) 摘要: 本发明实施例公开了一种硬件加速方法、编译器和设备, 用于提高代码执行效率从而实现硬件加速。本发明实施例方法包括: 编译器获取编译策略信息以及源代码; 所述编译策略信息指示第一代代码类型与第一处理器匹配, 第二代代码类型与第二处理器匹

配; 所述编译器根据所述编译策略信息分析所述源代码中的代码段, 确定属于第一代代码类型的第一代码段或属于第二代代码类型的第二代码段; 所述编译器将第一代代码段编译为第一可执行代码, 将所述第一可执行代码发往所述第一处理器; 将第二代代码段编译为第二可执行代码, 将所述第二可执行代码发往所述第二处理器。



WO 2017/000839 A1

**根据细则 4.17 的声明:**

— 关于申请人有权申请并被授予专利(细则 4.17(ii))

**本国际公布:**

— 包括国际检索报告(条约第 21 条(3))。

## 一种硬件加速方法、编译器以及设备

本申请要求于 2015 年 6 月 30 日提交中国专利局、申请号为 201510373054.7、发明名称为“一种硬件加速方法、编译器以及设备”的中国专利申请的优先权，其全部内容通过引用结合在本申请中。

5

### 技术领域

本发明涉及计算机技术领域，特别涉及一种硬件加速方法、编译器以及设备。

### 背景技术

10 源代码有时也称为源程序，是指未编译的按照一定的程序设计语言规范书写的程序指令序列。因此，源代码是一系列人类可读的计算机语言指令。在现代程序语言中，计算机源代码的最终目的是将人类可读的程序指令序列翻译成为计算机可以执行的二进制指令，这种过程叫做编译，通过编译器完成。

基于以上说明，目前代码编译以及执行流程如下：编译器在获得源代码后，将源代码编译为目标代码，然后将目标代码发送给执行代码的硬件如中央处理单元（英文：15 central processing unit，简称 CPU），由 CPU 执行目标代码获得执行结果。

但是以上方案代码执行效率较低。

### 发明内容

20 本发明实施例提供了一种硬件加速方法、编译器以及设备，用于提高代码执行效率从而实现硬件加速。

本发明实施例一方面提供了一种硬件加速方法，包括：

编译器获取编译策略信息以及源代码；所述编译策略信息指示第一代类型与第一处理器匹配，第二代类型与第二处理器匹配；

25 所述编译器根据所述编译策略信息分析所述源代码中的代码段，确定属于第一代类型的第一代代码段或属于第二代类型的第二代代码段；

所述编译器将第一代代码段编译为第一可执行代码，将所述第一可执行代码发往所述第一处理器；将第二代代码段编译为第二可执行代码，将所述第二可执行代码发往所述第二处理器。

30 结合一方面的实现方式，在第一种可选的实现方式中，所述将所述第二可执行代码

发往所述第二处理器包括：

所述编译器通过调度管理器将所述第二可执行代码发送给所述第二处理器；或者，

所述编译器在所述第二可执行代码中添加调用所述第二处理器的指示信息，通过调度管理器将所述第二可执行代码发送给所述第一处理器，使所述第一处理器获取到所述指示信息后将所述第二可执行代码发往所述第二处理器。

结合一方面的实现方式，在第二种可选的实现方式中，所述第一处理器为 CPU，第二处理器为 FPGA；所述编译器根据所述编译策略信息分析所述源代码中的代码段，确定属于第一代代码类型的第一代码段或属于第二代代码类型的第二代代码段包括：

所述编译器统计所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间，若统计得到的参数超过第一预定阈值，则确定该代码段属于第二代代码段，否则确定该代码段属于第一代代码段；或者，

所述编译器将所述源代码中的代码段与所述第二代代码段的特征规则进行比较，若与所述第二代代码段的特征规则匹配，则确定该代码段为第二代代码段，否则，确定该代码段为第一代代码段；或者，

所述编译器读取所述源代码中的加速标识信息，依据所述加速标识信息的指示确定所述源代码中的代码属于第一代代码段或第二代代码段。

结合一方面的第二种可选的实现方式，在第三种可选的实现方式中，所述编译器统计所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间包括：

所述编译器调用统计指令集通过所述源代码中的功能代码确定所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间；或者，所述编译器获取定时统计的代码执行统计报表确定所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间。

结合一方面、一方面的第一种、第二种或者第三种可选的实现方式，在第四种可选的实现方式中，若所述第二代代码段对应进程的优先级高于正在所述第二处理器中执行的可执行代码对应的进程的优先级，在将所述第二可执行代码发往所述第二处理器之前，所述方法还包括：

若所述第二处理器的繁忙度高于第二预定阈值，则停止正在所述第二处理器中执行的可执行代码；将正在所述第二处理器中执行的可执行代码对应的代码段编译为与第一处理器匹配的可执行代码，发送给所述第一处理器。

结合一方面、一方面的第一种、第二种或者第三种可选的实现方式，在第五种可选的实现方式中，若第一代代码段属于主代码段，所述第二代代码段从属所述主代码段，则所述方法还包括：

在所述第一可执行代码以及所述第二可执行代码中添加交互指令，使所述第二处理器向所述第一处理器返回所述第二可执行代码的执行结果。

本发明实施例二方面提供了一种硬件加速方法，包括：

编译器获取编译策略信息以及源代码；所述编译策略信息指示第一代代码类型与第一  
5 处理器匹配，第二代代码类型与第二处理器匹配；

所述编译器根据所述编译策略信息将所述源代码中的代码段编译为与第一处理器匹配的第一可执行代码，将所述第一可执行代码发往所述第一处理器；

若接收到所述第一处理器统计的所述第一可执行代码的第一执行信息，并依据所述  
10 第一执行信息确定所述第一可执行代码对应的代码段与第二处理器匹配，则将所述第一可执行代码对应的代码段编译为第二可执行代码，将所述第二可执行代码发往第二处理器。

结合二方面的实现方式，在第一种可选的实现方式中，在将所述第二可执行代码发往第二处理器之后，所述方法还包括：

接收来自所述第二处理器执行所述第二可执行代码的第二执行信息，若依据所述第  
15 二执行信息确定所述第二可执行代码对应的代码段与所述第二处理器匹配度低于期望值，则在所述第二处理器卸载所述第二可执行代码，将所述第二可执行代码对应的代码段对应的第一可执行代码发往所述第一处理器。

本发明实施例三方面还提供了一种编译器，包括：

获取单元，用于获取编译策略信息以及源代码；所述编译策略信息指示第一代代码类  
20 型与第一处理器匹配，第二代代码类型与第二处理器匹配；

确定单元，用于根据所述编译策略信息分析所述源代码中的代码段，确定属于第一代代码类型的第一代码段或属于第二代代码类型的第二代码段；

编译单元，用于将第一代代码段编译为第一可执行代码，将第二代代码段编译为第二可  
25 执行代码；

发送单元，用于将所述第一可执行代码发往所述第一处理器，将所述第二可执行代  
码发往所述第二处理器。

结合三方面的实现方式，在第一种可选的实现方式中，所述发送单元用于通过调度  
管理器将所述第二可执行代码发送给所述第二处理器；或者，

所述编译单元还用于在所述第二可执行代码中添加调用第二处理器的指示信息；

30 所述发送单元还用于将所述第二可执行代码发送给调度管理器，通过调度管理器将所述第二可执行代码发送给所述第一处理器，使所述第一处理器获取到所述指示信息后

将所述第二可执行代码发往所述第二处理器。

结合三方面的实现方式，在第二种可选的实现方式中，所述第一处理器为 CPU，第二处理器为 FPGA；所述确定单元用于统计所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间，若统计得到的参数超过第一预定阈值，则确定该代码段属于第二代码段，否则确定该代码段属于第一代代码段；或者，

所述确定单元用于将所述源代码中的代码段与所述第二代代码段的特征规则进行比较，若与所述第二代代码段的特征规则匹配，则确定该代码段为第二代代码段，否则，确定该代码段为第一代代码段；或者，

所述确定单元用于读取所述源代码中的加速标识信息，依据所述加速标识信息的指示确定所述源代码中的代码属于第一代代码段或第二代代码段。

结合三方面的第二种可选的实现方式，在第三种可选的实现方式中，所述确定单元用于调用统计指令集通过所述源代码中的功能代码确定所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间；或者，

所述确定单元用于获取定时统计的代码执行统计报表确定所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间。

结合三方面、三方面的第一种、第二种或者第三种可选的实现方式，在第四种可选的实现方式中，若所述第二代代码段对应进程的优先级高于正在所述第二处理器中执行的可执行代码对应的进程的优先级，所述设备还包括：

第一处理单元，用于若所述第二处理器的繁忙度高于第二预定阈值，则停止正在所述第二处理器中执行的可执行代码；

第二处理单元，用于将正在所述第二处理器中执行的可执行代码对应的代码段编译为与第一处理器匹配的可执行代码，发送给所述第一处理器。

结合三方面、三方面的第一种、第二种或者第三种可选的实现方式，在第五种可选的实现方式中，若第一代代码段属于主代码段，所述第二代代码段从属所述主代码段，所述编译器还包括：

第三处理单元，用于在所述第一可执行代码以及所述第二可执行代码中添加交互指令，使所述第二处理器向所述第一处理器返回所述第二可执行代码的执行结果。

本发明实施例四方面提供了一种编译器，包括：

获取单元，用于获取编译策略信息以及源代码；所述编译策略信息指示第一代代码类型与第一处理器匹配，第二代代码类型与第二处理器匹配；

第一编译单元，用于根据所述编译策略信息将所述源代码中的代码段编译为与第一

处理器匹配的第一可执行代码；

第一发送单元，用于将所述第一可执行代码发往所述第一处理器；

第一接收单元，用于接收所述第一处理器统计的所述第一可执行代码的第一执行信息；

- 5 第一处理单元，用于依据所述第一执行信息确定所述第一可执行代码对应的代码段是否与第二处理器匹配；

第二编译单元，用于所述第一处理单元依据所述第一执行信息确定所述第一可执行代码对应的代码段与第二处理器匹配，则将所述第一可执行代码对应的代码段编译为第二可执行代码；

- 10 第二发送单元，用于将所述第二可执行代码发往第二处理器。

结合四方面的实现方式，在第一种可选的实现方式中，所述编译器还包括：

第二接收单元，用于接收来自所述第二处理器执行所述第二可执行代码的第二执行信息；

- 15 第二处理单元，用于若依据所述第二执行信息确定所述第二可执行代码对应的代码段与所述第二处理器匹配度低于期望值，则在所述第二处理器卸载所述第二可执行代码，将所述第二可执行代码对应的代码段对应的第一可执行代码发往所述第一处理器。

本发明实施例五方面提供了一种设备，包括：

编译器和调度管理器，所述编译器包括：

获取单元、确定单元、编译单元以及发送单元；

- 20 所述获取单元，用于获取编译策略信息以及源代码；所述编译策略信息指示第一代码类型与第一处理器匹配，第二代码类型与第二处理器匹配；

所述确定单元，用于根据所述编译策略信息分析所述源代码中的代码段，确定属于第一代码类型的第一代码段或属于第二代码类型的第二代码段；

- 25 所述编译单元，用于将第一代码段编译为第一可执行代码，将第二代码段编译为第二可执行代码；

所述发送单元，用于将所述第一可执行代码和所述第二可执行代码发送给调度管理器；

所述调度管理器，用于将所述第一可执行代码发往所述第一处理器，将所述第二可执行代码发往所述第二处理器。

- 30 结合五方面的实现方式，在第一种可选的实现方式中，所述调度管理器，还用于配置编译策略信息，将所述编译策略信息发送给所述编译器。

结合五方面的实现方式，在第二种可选的实现方式中，若所述第一处理器为 CPU，所述第二处理器为 FPGA，

所述确定单元，还用于调用统计指令集通过所述源代码中的功能代码确定所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间；或者

5 所述调度管理器，还用于获取定时统计的代码执行统计报表，将所述定时统计的代码执行统计报表发送给所述编译器；

所述确定单元，还用于依据所述代码执行统计报表确定所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间；

所述确定单元用于确定属于第一代码类型的第一代码段或属于第二代码类型的第二代码段包括：若所述循环执行次数和/或 CPU 的执行时间超过第一预定阈值，则确定该代码段属于第二代码段，否则确定该代码段属于第一代码段。

结合五方面的实现方式，在第三种可选的实现方式中，所述调度管理器，还用于若所述第二处理器的繁忙度高于第二预定阈值，在所述编译器停止正在所述第二处理器中执行的可执行代码，将正在所述第二处理器中执行的可执行代码对应的代码段编译为与第一处理器匹配的可执行代码后，接收来自所述与第一处理器匹配的可执行代码，并发送给第一处理器。

从以上技术方案可以看出，本发明实施例具有以下优点：编译器将源代码中与第一处理器匹配的代码段编译成第一处理器可以执行的第一可执行代码，将源代码中与第二处理器匹配的代码段编译成第二处理器可以执行的第二可执行代码；第一可执行代码被发往了第一处理器，第二可执行代码被发往了第二处理器，因此源代码中的代码段对应的可执行代码被分配到匹配度较高的硬件来执行，因此执行效率会更高，实现了硬件加速。

## 附图说明

25 为了更清楚地说明本发明实施例中的技术方案，下面将对实施例描述中所需要使用的附图作简要介绍。

图 1 为本发明实施例中系统架构示意图；

图 2 为本发明实施例方法流程示意图；

图 3 为本发明实施例方法流程示意图；

30 图 4 为本发明实施例方法流程示意图；

图 5 为本发明实施例方法流程示意图；



- 图 6 为本发明实施例方法流程示意图；  
图 7 为本发明实施例方法流程示意图；  
图 8 为本发明实施例方法流程示意图；  
图 9 为本发明实施例编译器结构示意图；  
5 图 10 为本发明实施例编译器结构示意图；  
图 11 为本发明实施例编译器结构示意图；  
图 12 为本发明实施例编译器结构示意图；  
图 13 为本发明实施例设备结构示意图；  
图 14 为本发明实施例设备结构示意图；  
10 图 15 为本发明实施例服务器结构示意图。

### 具体实施方式

为了使本发明的目的、技术方案和优点更加清楚，下面将结合附图对本发明作进一步地详细描述。

- 15 本发明实施例提供了一种硬件加速方法、编译器以及设备，用于提高代码执行效率从而实现硬件加速。

为了方便理解本发明实施例，下面介绍本发明实施例中的系统架构。

请参阅图 1，该系统架构包括编译器、调度管理器、第一处理器、第二处理器以及程序库。

- 20 其中调度管理器负责管理和调度一片或多片第一处理器和第二处理器；把可执行文件加载到第一处理器和第二处理器上，把可执行文件从第一处理器和第二处理器上卸载；负责向编译器提供编译策略和硬件信息。

编译器用于编译源代码、生成中间代码和可执行代码。

- 25 第一处理器为 CPU、图形处理单元（graphics processing unit，GPU）、网络处理器（network processor，NP）等芯片。

第二处理器为现场可编程门阵列（field-programmable gate array，FPGA）、复杂可编程逻辑设备（complex programmable logic device，CPLD）、可编程阵列逻辑（programmable array logic，PAL）和通用阵列逻辑（generic array logic，GAL）等可编程逻辑器件。

- 30 程序库中包括：源代码，中间代码和可执行代码。

在上述系统架构的基础上，对本发明提供的硬件加速方法通过以下实施例进行说

明：

请参阅图 2，本发明实施例中一种硬件加速方法包括：

101、编译器获取编译策略信息以及源代码；该编译策略信息指示第一代码类型与第一处理器匹配，第二代码类型与第二处理器匹配；

5 本实施例中，编译器可以通过多种方法获取编译策略信息，例如：从本地存储器中获取预配置的编译策略信息，或者，接收调度管理器下发的编译策略信息。

在本实施例中，本地存储器中可以存储编译策略信息，也可以存储编译策略信息的指示信息，例如：指示如何获取编译策略信息的信息；或者编译策略信息的地址信息等。

该编译策略信息包括：编译检测出可以用 FPGA 进行加速时采用的策略，编译 FPGA  
10 加速效果阈值，编译最长时间阈值（指由于 FPGA 的编译有时候花费的时间比较长，设置一个编译最长时间），编译异常策略（指在 FPGA 编译过程中和 CPU 编译过程中可能会产生一系列的异常，通过设置异常策略来处理这些异常）。

上述 CPU 用于指代第一处理器，上述 FPGA 用于指代第二处理器。

代码类型包括：可加速代码以及普通代码。其中可加速代码可以为：CPU 指令，代  
15 码块，函数（模块）以及源文件。

第一处理器和第二处理器分别为：CPU、FPGA、GPU、NP、专用集成电路（application-specific integrated circuit, ASIC）、CPLD、PAL 和 GAL 中的任意两项。可以理解的是，CPU，GPU，NP 为具有软件编程能力的芯片，FPGA，ASIC，PAL，GAL，CPLD 为可编程逻辑器件。

20 源代码可以由一个或多个文件组成，源代码所使用的编程语言可以是汇编语言，C 语言以及脚本编程语言，还包括其他类型的语言，此处不再赘述。

102、该编译器根据该编译策略信息分析该源代码中的代码段，确定属于第一代码类型的第一代码段或属于第二代码类型的第二代码段；

在获取编译策略信息以及源代码后，编译器分析该源代码中的代码段，确定属于第  
25 一代码类型的第一代码段或属于第二代码类型的第二代码段；

其中，该编译器会根据编译策略信息检测源代码中是否有可以加速的代码，根据检测结果将源代码中的普通代码生成中间源代码。中间源代码可以理解为源代码中的普通代码的另一种形式存在的源代码。

当第一处理器为 CPU，第二处理器为 FPGA 时，可选的，该编译器统计该源代码中的  
30 的代码段的循环执行次数和/或 CPU 的执行时间，若统计得到的参数超过第一预定阈值，则确定该代码段属于第二代码段，否则确定该代码段属于第一代码段；或者，该编译器

将该源代码中的代码段与该第二代码段的特征规则进行比较，若与该第二代码段的特征规则匹配，则确定该代码段为第二代码段，否则，确定该代码段为第一代码段；或者，该编译器读取该源代码中的加速标识信息，依据该加速标识信息的指示确定该源代码中的代码属于第一代码段或第二代码段。其中该编译器统计该源代码中的代码段的循环执行次数和/或 CPU 的执行时间具体为：该编译器调用统计指令集通过该源代码中的功能代码确定该源代码中的代码段的循环执行次数和/或 CPU 的执行时间；或者，该编译器获取定时统计的代码执行统计报表确定该源代码中的代码段的循环执行次数和/或 CPU 的执行时间。统计指令集可以由 CPU 扩展，统计指令集可以对执行代码进行统计，编译器在编译时自动调用统计指令集，定时统计的代码执行统计报表可以通过外部程序实现，或者人工干预代码实现。

需要说明的是，上述第一预定阈值是与统计得到的参数对应的，若统计得到的参数很多，那么第一预定阈值可以是一组参数值；例如：统计得到的是循环执行次数，那么上述第一预定阈值可以是诸如 100 次、50 次等参数，那么只要循环执行次数超过这个第一预定阈值指定的次数，那么表示需要将代码段编译为 FPGA 的可执行代码；若统计得到的是循环执行次数和 CPU 执行时间，那么可以是一组参数值，例如：100 次、2S；那么循环执行次数超过 100 次，CPU 执行时间超过 2S 都会出现超过阈值的情况，此种情况下，可以认为超过任一参数都认为超过第一预定阈值，也可以必须满足超过了以上两个参数才认为超过第一预定阈值。基于统计得到的参数不同，第一预定阈值还可以有其他参数类型，第一预定阈值的具体参数值可以按照经验或者加速需求进行确定，以上举例不应理解为对本发明实施例的唯一性限定。

另外，上述特征规则指的是编译器预先设置一个可加速代码描述特征库，可加速代码描述特征库中包含着特征规则。

进一步的，上述加速标识信息指的是编程人员在源代码中选取目标代码，给目标代码添加标识，用来表示该目标代码可被加速。

103、该编译器将第一代码段编译为第一可执行代码，将该第一可执行代码发往上述第一处理器；将第二代码段编译为第二可执行代码，将该第二可执行代码发往上述第二处理器。

该编码器将该第一代码编译为第一处理器可执行代码，即第一可执行代码，将该第一可执行代码发往上述第一处理器；将该第二代码段编译为第二处理器可执行代码，即第二可执行代码，将该第二可执行代码发往上述第二处理器。在本发明实施例中，编译器将可执行代码“发往”处理器，不应狭隘的理解为以数据包的方式进行发送，应广义理

解其含义为数据传递方式；数据传递方式除了以数据包的方式发送外，在编译器和处理器之间通常可以使用的数据传递方式还有很多，例如：将可执行代码存到存储空间内，例如：磁盘内，然后由 CPU 读取该执行代码；或者，将可执行代码存储到磁盘内，然后下载到 FPGA 芯片中。具体的数据传递方式可以依硬件的需求进行选用，本实施例对此不进行限定。

另外，若第一或者第二可执行代码的执行量很多，并且存在多个第一硬件或第二硬件的情况下，可以在同类硬件之间进行负载均衡处理。负载均衡的方式可以参考各种已经存在的负载均衡算法，本实施例不再赘述。

可以理解的是，在第一处理器和第二处理器分别接收到第一可执行代码和第二可执行代码后，第一处理器和第二处理器通过交互指令完成代码的交互。

可选的，在将该第二可执行代码发往该第二处理器之前，若该第二代码段对应进程的优先级高于正在该第二处理器中执行的可执行代码对应的进程的优先级，则有：若该第二处理器的繁忙度高于第二预定阈值，则停止正在该第二处理器中执行的可执行代码；将正在该第二处理器中执行的可执行代码对应的代码段编译为与第一处理器匹配的可执行代码，发送给该第一处理器。

需要说明的是，上述繁忙度指的是第二处理器的数据处理资源被占用的程度，占用得越多则表示繁忙度越高；如果出现拥塞排队等情况，则排队等待时间越长繁忙度越高。第二预定阈值是繁忙程度的临界值，例如资源占用率超过某一阈值或者排队时间超过某一阈值等，这时可以确定第二处理器处于繁忙状态。另外，上述优先级指的是为不同的执行逻辑（任务、进程或线程等）分配不同的优先级。

可选的，若第一代码段属于主代码段，该第二代码段从属该主代码段，则有：在该第一可执行代码以及该第二可执行代码中添加交互指令，使该第二处理器向该第一处理器返回该第二可执行代码的执行结果。

上述主代码段可以理解为普通代码段，从代码段可以理解为可加速代码段。

另外，编译器可以通过多种方法将第二可执行代码发往该第二处理器，例如：通过调度管理器将该第二可执行代码发送给该第二处理器；或者，在该第二可执行代码中添加调用第二处理器的指示信息，通过调度管理器将该第二可执行代码发送给该第一处理器，使该第一处理器获取到该指示信息后将该第二可执行代码发往该第二处理器。

本实施例中，编译器将源代码中与第一处理器匹配的代码段编译成第一处理器可以执行的第一可执行代码，将源代码中与第二处理器匹配的代码段编译成第二处理器可以执行的第二可执行代码；第一可执行代码被发往了第一处理器，第二可执行代码被发往

了第二处理器，因此源代码中的代码段对应的可执行代码被分配到匹配度较高的硬件来执行，因此执行效率会更高，实现了硬件加速。

请参阅图 3，本发明实施例中一种硬件加速方法包括：

201、编译器获取编译策略信息以及源代码；该编译策略信息指示第一代码类型与  
5 第一处理器匹配，第二代码类型与第二处理器匹配；

本实施例中，步骤 201 和步骤 101 类似，此处不再赘述。

202、该编译器根据该编译策略信息将该源代码中的代码段编译为与第一处理器匹  
配的第一可执行代码，将该第一可执行代码发往该第一处理器；

该编译器通过编译器中的编译单元将该源代码中的代码段编译为与第一处理器匹  
10 配的第一可执行代码，将该第一可执行代码发往该第一处理器。

可以理解的是，编译器可以根据编译策略信息，生成中间源代码，再对中间源代码  
中的代码段进行编译。

203、若接收到该第一处理器统计的该第一可执行代码的第一执行信息，并依据该  
第一执行信息确定该第一可执行代码对应的代码段与第二处理器匹配，则将该第一可执  
15 行代码对应的代码段编译为第二可执行代码，将该第二可执行代码发往第二处理器。

第一处理器接收到第一可执行代码后，生成了第一执行信息，并将该第一执行信息  
发往编译器，编译器依据该第一执行信息确定该第一可执行代码对应的代码段与第二处  
理器匹配，则编译器通过调度管理器修改编译策略信息，将该第一可执行代码对应的代  
码段编译为第二可执行代码，并通过调度管理器将该第二可执行代码发往第二处理器。

20 上述第一执行信息是用来确定可执行代码与处理器之间是否匹配的参数，基于前述  
举例，这里的执行信息可以是第一可执行代码在第一处理器中的执行次数或执行时间，  
还可以是其他信息，在此处不再一一举例说明。

可选的，步骤 203 之后，还可以包括：

接收来自该第二处理器执行该第二可执行代码的第二执行信息，若依据该第二执行  
25 信息确定该第二可执行代码对应的代码段与该第二处理器匹配度低于期望值，则在该第  
二处理器卸载该第二可执行代码，将该第二可执行代码对应的代码段对应的第一可执行  
代码发往该第一处理器。

需要说明的是，第二可执行代码对应的代码段对应的第一可执行代码指的是将第二  
可执行代码对应的代码段重新编译，生成第一可执行代码，将第一可执行代码发往第一  
30 处理器。

在本实施例中，第二执行信息也是用来确定可执行代码与处理器之间是否匹配的参

数，可以参考第一执行信息选用的参数举例说明。

本实施例中，将源代码中的代码段编译为与第一处理器匹配的第一可执行代码，当确定第一可执行代码对应的代码段与第二处理器匹配时，将第一可执行代码对应的代码段编译为第二可执行代码，通过第二处理器接收第二可执行代码，因此源代码中的代码段对应的可执行代码被分配到匹配度较高的硬件来执行，因此执行效率会更高，实现了硬件加速。

此外，当第二可执行代码在第二处理器执行效率不高时，则在第二处理器上卸载该第二可执行代码，并对第二可执行代码对应的代码段重新编译，通过第一处理器来执行。

为了便于理解，下面对本发明实施例中的硬件加速方法进行详细描述，请参阅图 4，本发明实施例中一种硬件加速方法包括：

301、编译器接收调度管理器下发的编译策略信息和硬件信息；

本实施例中，调度管理器配置编译策略信息和硬件信息，调度管理器将该编译策略信息和硬件信息发送给编译器。

本实施例中的编译策略信息为将可加速代码直接编译成 FPGA 可执行代码并在 FPGA 中加载运行。

硬件信息包括：CPU 的型号和数量、CPU 上的操作系统类型和版本号、CPU 可使用的内存信息、FPGA 型号和数量、与 FPGA 相连的外围芯片信息以及 CPU 和 FPGA 之间的交互方式。

302、编译器接收源代码；

源代码可以由一个或多个文件组成，源代码所使用的编程语言可以是汇编语言，C 语言以及脚本编程语言，还包括其他类型的语言，此处不再赘述。以 C 语言为例，假设本实施例中的源代码为：

```
int func1(int a[8])
{
    int a[8], b;
    ...
    b = a[0]*a[1]*a[2]*a[3]*a[4]*a[5]*a[6]*a[7];
    return b;
}
```

303、编译器分析该源代码，确定可加速代码；

编译器中预先设置一个可加速代码描述特征库，编译器将源代码与该可加速代码描

述特征库中的特征规则进行比较，如果源代码中有某段代码能匹配上，则表明上述某段代码可以用 FPGA 来加速。

需要说明的是，还可以采用其他方法，例如：编程人员可以在源代码中设置需要加速的代码，比如：

```

5      #program fpga_acc_func
      int func1(int a[8])
      {

          int a[8], b;

10         b = a[0]*a[1]*a[2]*a[3]*a[4]*a[5]*a[6]*a[7];

          return b;
      }

```

15 用#program fpga\_acc\_func 来表明该函数需要被加速。又比如：

```

      int func1(int a[8])
      {

          int a[8], b;

20         #program fpga_acc_begin
          b = a[0]*a[1]*a[2]*a[3]*a[4]*a[5]*a[6]*a[7];
          #program fpga_acc_end

25         return b;
      }

```

用#program fpga\_acc\_begin 和#program fpga\_acc\_end 来表明该对之间的代码需要被加速。

304、编译器将源代码中的普通代码转换为中间源代码；

30 编译器根据可加速代码描述特征库、编译策略和硬件信息来检测源代码中是否有可以加速的代码，然后根据检测结果将源代码中的普通代码转换为中间源代码，中间源代

码可以为：

```
int func1(int a[8])
{
```

```
5     int a[8], b;
    ...
    b = fpga_multiply(a);
    return b;
}
```

10 305、编译器将中间源代码编译成 CPU 可执行代码，并向调度管理器发送 CPU 可执行代码；

编译器通过 CPU 编译单元将中间源代码编译成 CPU 可执行代码，CPU 编译单元位于编译器中。

15 306、编译器将可加速代码编译成 FPGA 可执行代码，并向调度管理器发送 FPGA 可执行代码；

编译器通过 FPGA 编译单元将可加速代码编译成 FPGA 可执行代码，并向调度管理器发送 FPGA 可执行代码，FPGA 编译单元位于编译器中。

需要说明的是，可加速代码在 FPGA 中的执行效率高于在 CPU 中的执行效率。

20 另外，也可以将可加速代码转换为逻辑语言代码，在将转换后的逻辑语言代码编译成 FPGA 可执行代码。

307、调度管理器将 CPU 可执行代码加载到 CPU 上执行；

调度管理器接收来自编译器的 CPU 可执行代码，调度管理器将 CPU 可执行代码加载到 CPU 上执行。

308、调度管理器将 FPGA 可执行代码加载到 FPGA 上执行；

25 调度管理器接收来自编译器的 FPGA 可执行代码，调度管理器将 FPGA 可执行代码加载到 FPGA 上执行。

需要说明的是，步骤 307 可以位于步骤 308 后面，步骤 307 与步骤 308 之间的顺序可以调换。

309、CPU 和 FPGA 执行代码的交互。

30 可以通过预配置的互操作协议来实现 CPU 可执行代码和 FPGA 可执行代码的交互，具体地，可以定义互操作原语：



Handshake, 握手。

Negotiate, 协商能力, 包括各自支持的能力和规格, 使用的互操作协议版本和特性等。

WriteData, 直接向对方写数据。

5 ReadData, 直接向对方读数据。

WriteDataAddress, 通过地址向对方写数据。比如在 DMA 方式下, 通过对方告诉的一个内存地址和长度, 向该地址中写内容。

ReadDataAddress, 通过地址向对方读数据。比如在 DMA 方式下, 通过对方告诉的一个内存地址和长度, 向该地址中读内容。

10 Call, 调用对方的功能。

Callback, 调用对方的功能, 该功能执行后的返回值或者功能回调。

本实施例中, 编译器将源代码中与 FPGA 匹配的代码段编译成 FPGA 可执行代码, 将源代码中与 CPU 匹配的代码段编译成 CPU 可执行代码; FPGA 可执行代码被发往了 FPGA, CPU 可执行代码被发往了 CPU, 因此源代码中的代码段对应的可执行代码被分  
15 配到匹配度较高的硬件来执行, 因此执行效率会更高, 实现了硬件加速。

为了便于理解, 下面对本发明实施例中的硬件加速方法进行详细描述, 请参阅图 5, 本发明实施例中一种硬件加速方法包括:

401、编译器接收调度管理器下发的编译策略信息和硬件信息;

本实施例中, 编译策略信息为: 统计源代码中的可加速代码在 CPU 中的执行次数, 当执行次数达到第一预定阈值 (本实施例假设第一预定阈值为 100) 后, 将可加速代码  
20 编译成 FPGA 可执行代码并在 FPGA 中加载运行。

402、编译器接收源代码;

假设本实施例中的源代码为:

```
void func2(bool c, FILE* file1, FILE* file2)
25 {
    if (c)
    {
        zip_func(file1);
        unzip_func(file2);
30     }
}
```

403、编译器分析该源代码，确定可加速代码；

步骤 403 与步骤 303 类似，此处不再赘述。

404、编译器将源代码中的普通代码转换为中间源代码；

中间源代码可以为：

```

5 void func2(bool c, FILE* file1, FILE* file2)
  {
    if (c)
      {
        static int cnt=0;
10      if (cnt++ < 100)
          {
            zip_func(file1);
            unzip_func(file2);
          }
15      else
        {
          fpga_zip_func(file1);
          fpga_unzip_func(file2);
        }
20    }
  }

```

405、编译器将中间源代码编译成 CPU 可执行代码，并向调度管理器发送 CPU 可执行代码；

406、编译器将可加速代码编译成 FPGA 可执行代码，并向调度管理器发送 FPGA  
25 可执行代码，FPGA 可执行代码携带调用 FPGA 的指示信息；

步骤 405、步骤 406 与步骤 305、步骤 306 类似，此处不再赘述。

另外，步骤 405 可以位于步骤 406 后面，步骤 405 与步骤 406 之间的顺序可以调换。

此外，编译器在 FPGA 可执行代码中添加调用 FPGA 的指示信息。其中指示信息表示：当 FPGA 可执行代码在 CPU 中的执行次数大于 100 次（说明执行效率过低），则将  
30 FPGA 可执行代码发往 FPGA。

407、调度管理器将 CPU 可执行代码加载到 CPU 上执行，将 FPGA 可执行代码加

载到 CPU 上执行；

408、CPU 根据指示信息将 FPGA 可执行代码发往 FPGA；

在确定 FPGA 可执行代码在 CPU 中的执行次数大于 100 次后，CPU 根据指示信息将 FPGA 可执行代码发往 FPGA。

5 409、CPU 和 FPGA 执行代码的交互。

本实施例中，首先将 FPGA 可执行代码和 CPU 可执行代码发送给 CPU，CPU 根据 FPGA 可执行代码携带的指示信息以及 FPGA 可执行代码在 CPU 中的执行次数大于第一预定阈值，将 FPGA 可执行代码发往 FPGA，由于 FPGA 可执行代码在 FPGA 中执行效率更高，因此执行效率会更高，实现了硬件加速。

10 为了便于理解，下面对本发明实施例中的硬件加速方法进行详细描述，请参阅图 6，本发明实施例中一种硬件加速方法包括：

501、编译器接收调度管理器下发的编译策略信息和硬件信息；

502、编译器接收源代码；

503、编译器分析该源代码，确定可加速代码；

15 504、编译器将源代码中的普通代码转换为中间源代码；

505、编译器将中间源代码编译成第一 CPU 可执行代码，并向调度管理器发送第一 CPU 可执行代码；

506、编译器将可加速代码编译成第一 FPGA 可执行代码，并向调度管理器发送第一 FPGA 可执行代码；

20 步骤 501 至步骤 506 与步骤 401 至步骤 406 类似，此处不再赘述。

507、调度管理器将第一 CPU 可执行代码加载到 CPU 上执行；

508、调度管理器将第一 FPGA 可执行代码加载到 FPGA 上执行；

需要说明的是，步骤 507 可以位于步骤 508 后面，步骤 507 与步骤 508 之间的顺序可以调换。

25 509、CPU 确定第一 CPU 可执行代码中各代码的执行次数；

第一 CPU 可执行代码中可能存在执行效率较低的代码，这些执行效率低的代码执行次数也就高，故可以通过执行次数找出可加速代码。

510、CPU 向调度管理器上报执行次数的统计信息；

30 当第一 CPU 可执行代码中存在代码的执行次数大于 100 次时，说明该代码可以被加速，可以由 FPGA 来执行该代码，因此 CPU 向调度管理器上报统计信息，统计信息为该代码的执行次数；需要说明的是，FPGA 也有可能向调度管理器上报统计信息（此

时在 FPGA 中存在执行效率低的代码)。

511、调度管理器修改编译策略信息，并将修改后的编译策略信息发往编译器；

调度管理器接收到统计信息后，确定执行次数大于 100 次，调度管理器修改编译策略信息，并将修改后的编译策略信息发往编译器。

5 512、编译器根据编译策略信息重新编译，生成第二 CPU 可执行代码和第二 FPGA 可执行代码，向调度管理器发送第二 CPU 可执行代码和第二 FPGA 可执行代码；

可以理解的是，编译器将源代码重新编译生成中间源代码，显然，此处的中间源代码和步骤 404 中的源代码是不同的。

513、调度管理器将第二 CPU 可执行代码加载到 CPU 上执行；

10 514、调度管理器将第二 FPGA 可执行代码加载到 FPGA 上执行；

515、CPU 和 FPGA 执行代码的交互。

步骤 513 至步骤 515 与步骤 307 至步骤 309 类似，此处不再赘述。

本实施例中，当 CPU 可执行代码中存在代码的执行次数大于第一预定阈值时，说明执行速度过慢，所以通过调度管理器修改编译策略信息，使编译器重新编译，生成新的 CPU 可执行代码和新的 FPGA 可执行代码，因此执行效率会更高，实现了硬件加速。

15 为了便于理解，下面对本发明实施例中的硬件加速方法进行详细描述，请参阅图 7，本发明实施例中一种硬件加速方法包括：

601、编译器接收调度管理器下发的编译策略信息和硬件信息；

602、编译器接收源代码；

20 603、编译器分析该源代码，确定可加速代码；

604、编译器将源代码转换为中间源代码；

605、编译器将中间源代码编译成第一 CPU 可执行代码，并向调度管理器发送第一 CPU 可执行代码；

步骤 601 至步骤 605 与步骤 401 至步骤 405 类似，此处不再赘述。

25 另外，可以理解的是，源代码中的可加速代码也被编译成第一 CPU 可执行代码。

606、调度管理器将第一 CPU 可执行代码加载到 CPU 上执行；

607、CPU 确定可加速代码在 CPU 中的执行次数；

可加速代码也被编译成第一 CPU 可执行代码，显然，可加速代码在 CPU 中的执行效率不高，也就是执行次数会比普通代码执行次数更多。

30 608、CPU 向调度管理器上报执行次数的统计信息；

当执行次数大于 100 时，说明该可加速代码可以由 FPGA 来执行，来提高执行效率，

因此 CPU 向调度管理器上报执行次数的统计信息。

609、调度管理器修改编译策略信息，并将修改后的编译策略信息发往编译器；

当调度管理器通过统计信息确定可加速代码在 CPU 中的执行效率不高时，调度管理器修改编译策略信息，并将修改后的编译策略信息发往编译器。

5       610、编译器根据修改后的编译策略信息将源代码中的普通代码重新生成中间源代码；

611、编译器将重新生成的中间源代码编译成第二 CPU 可执行代码，并发送给调度管理器；

612、编译器将可加速代码编译成 FPGA 可执行代码，并发送给调度管理器；

10       613、调度管理器将第二 CPU 可执行代码加载到 CPU 上执行；

614、调度管理器将 FPGA 可执行代码加载到 FPGA 上执行；

615、CPU 和 FPGA 执行代码的交互。

本实施例中，编译器将源代码转换为中间源代码，再将中间源代码编译为 CPU 可执行代码（同时，未编译 FPGA 可执行代码），当 CPU 可执行代码中存在代码的执行次数大于第一预定阈值时，说明该代码可以由 FPGA 来执行，来提高执行效率，通过重新编译的方法，生成 FPGA 可执行代码和新的 CPU 可执行代码，因此执行效率会更高，实现了硬件加速。

15

需要说明的是，上述各实施例中，编译策略信息还可以为：

一定时间内代码在 CPU 中的执行次数达到第一预定阈值，则编译成 FPGA 可执行代码并加载运行。或者统计代码的执行时间，当执行时间超过第一预定阈值，则编译成 FPGA 可执行代码并加载运行。

20

另外，统计源代码中的代码在 CPU 中的执行次数可以在编译阶段由编译器智能插入统计功能代码，也可以在 CPU 中扩展统计指令集，通过统计指令集对代码进行统计，编译器在编译时自动调用扩展统计指令集。进一步的，也可以通过外部程序或人工干预的方法。

25

进一步的，当 FPGA 接收到 FPGA 可执行代码，但是 FPGA 中没有足够的资源来执行加速，则判断该 FPGA 可执行代码对应的进程 1 的优先级是否高于正在 FPGA 中执行的可执行代码对应的进程 2 的优先级（优先级可以根据进程预先设置），若判断是，则将进程 2 在 FPGA 中加速的功能占用的资源释放出来，由进程 1 来使用。

30       为了便于理解，下面对本发明实施例中的硬件加速方法进行详细描述，请参阅图 8，本发明实施例中另一种硬件加速方法包括：

701、CPU 确定 FPGA 中 FPGA 可执行代码的执行次数；

需要说明的是，FPGA 可执行代码的执行次数还可以由 FPGA 本身统计得到。

步骤 701 之前，已经有可执行代码分别在 CPU 和 FPGA 中运行，并且原编译策略信息为：在一段时间内统计 FPGA 可执行代码在 FPGA 中的执行次数，当执行次数小于  
5 第一预定阈值（100）时，将 FPGA 可执行代码对应的代码重新编译为 CPU 可执行代码并在 CPU 中加载运行。

702、CPU 向调度管理器上报执行次数的统计信息；

703、调度管理器修改编译策略信息，并将修改后的编译策略信息发往编译器；

当统计信息中的执行次数小于 100 时，修改编译策略信息，修改的编译策略信息为：  
10 不使用 FPGA 加速，重新编译和加载。

704、编译器根据修改后的编译策略信息重新将 FPGA 可执行代码对应的源代码转换为新的中间源代码；

假设源代码为：void func2(bool c, FILE\* file1, FILE\* file2)

```

{
15     if (c)
        {
            zip_func(file1);
            unzip_func(file2);
        }
20 }

```

步骤 701 之前，源代码转换的中间源代码可以为：

```

void func2(bool c, FILE* file1, FILE* file2)
{
    if (c)
25     {
        static int cnt=0;
        static int fpga_cnt=0;
        if (cnt++ < 100)
        {
30             zip_func(file1);
            unzip_func(file2);

```

```

        }
        else
        {
            fpga_zip_func(file1);
5           fpga_unzip_func(file2);
        fpga_cnt++;
        report_to_manager(fpga_cnt);
        }
    }
10 }

```

则编译器根据修改后的编译策略信息，重新生成的中间源代码可以为：

```

void func2(bool c, FILE* file1, FILE* file2)
{
    if (c)
15     {
        static int cnt=0;

        zip_func(file1);
        unzip_func(file2);

20         if (cnt++ >= 100)
            {
                report_to_manager(cnt);
            }
25     }
}

```

705、编译器将新的中间源代码编译成 CPU 可执行代码，并发送给调度管理器；

706、调度管理器将 CPU 可执行代码加载到 CPU 上执行；

707、调度管理器将 FPGA 可执行代码从 FPGA 中卸载。

30 本实施例中，当 FPGA 中执行的代码在一定时间内执行次数小于第一预定阈值时，说明此时 FPGA 执行效率不高，故重新编译，通过 CPU 来执行，因此能够提高执行效

率，实现硬件加速。

为了便于理解，下面以一实际的应用场景对本发明实施例中的硬件加速方法进行描述：

调度管理器配置编译策略信息以及硬件信息（CPU 和 FPGA），调度管理机将该编译策略信息以及硬件信息发送给编译器，编译器接收源代码，编译器预先设置一个可加速代码特征库，将源代码和该可加速代码特征库中的特征规则进行比较，该特征规则为：如果源代码中存在匹配的代码段，则表明该代码段可以用 FPGA 加速。在编译器通过特征规则确定了属于普通代码类型的第一代码段，以及属于可加速代码类型的第二代码段后，将该第一代码段编译为 CPU 可执行代码，通过调度管理器发送给 CPU，将该第二代码段编译为 FPGA 可执行代码，通过调度管理器发送给 FPGA。

上面对本发明实施例中的硬件加速方法进行了描述，下面对本发明实施例中的编译器进行描述，请参阅图 9，本发明实施例编译器包括：

获取单元 801，用于获取编译策略信息以及源代码；该编译策略信息指示第一代码类型与第一处理器匹配，第二代码类型与第二处理器匹配；

15 确定单元 802，用于根据该编译策略信息分析该源代码中的代码段，确定属于第一代码类型的第一代码段或属于第二代码类型的第二代码段；

编译单元 803，用于将第一代码段编译为第一可执行代码，将第二代码段编译为第二可执行代码；

20 发送单元 804，用于将该第一可执行代码发往该第一处理器，将该第二可执行代码发往该第二处理器。

可选的，上述获取单元 801，用于从本地存储器中获取预配置的编译策略信息；或者，获取单元 801，用于接收调度管理器下发的编译策略信息。

其中第一处理器和第二处理器分别为：CPU，FPGA，GPU，NP，ASIC，CPLD，PAL 和 GAL 中的任意两项。可以理解的是，CPU，GPU，NP 为具有软件编程能力的芯片，FPGA，ASIC，PAL，GAL，CPLD 为可编程逻辑器件。

若该第二代码段对应进程的优先级高于正在该第二处理器中执行的可执行代码对应的进程的优先级，可选的，本实施例还包括：

第一处理单元 805，用于若该第二处理器的繁忙度高于第二预定阈值，则停止正在该第二处理器中执行的可执行代码；

30 第二处理单元 806，用于将正在该第二处理器中执行的可执行代码对应的代码段编译为与第一处理器匹配的可执行代码，发送给该第一处理器。



可选的，上述发送单元 804，用于通过调度管理器将该第二可执行代码发送给该第二处理器；或者，编译单元 803 还用于在该第二可执行代码中添加调用第二处理器的指示信息；发送单元 804 还用于将第二可执行代码发送给调度管理器，通过调度管理器将该第二可执行代码发送给该第一处理器，使该第一处理器获取到该指示信息后将该第二可执行代码发往该第二处理器。

当第一处理器为 CPU，第二处理器为 FPGA 时，可选的，确定单元 802 用于统计该源代码中的代码段的循环执行次数和/或 CPU 的执行时间，若统计得到的参数超过第一预定阈值，则确定该代码段属于第二代码段，否则确定该代码段属于第一代码段；或者，

确定单元 802 用于将该源代码中的代码段与第二代码段的特征规则进行比较，若与该第二代码段的特征规则匹配，则确定该代码段为第二代码段，否则，确定该代码段为第一代码段；或者，

确定单元 802 用于读取该源代码中的加速标识信息，依据该加速标识信息的指示确定该源代码中的代码属于第一代码段或第二代码段；或者，

确定单元 802 用于调用统计指令集通过该源代码中的功能代码确定该源代码中的代码段的循环执行次数和/或 CPU 的执行时间；或者，

确定单元 802 用于获取定时统计的代码执行统计报表确定该源代码中的代码段的循环执行次数和/或 CPU 的执行时间。

可选的，本实施例还包括：第三处理单元 807，用于在该第一可执行代码以及该第二可执行代码中添加交互指令，使该第二处理器向该第一处理器返回该第二可执行代码的执行结果。

本实施例中，编译器将源代码中与第一处理器匹配的代码段编译成第一处理器可以执行的第一可执行代码，将源代码中与第二处理器匹配的代码段编译成第二处理器可以执行的第二可执行代码；第一可执行代码被发往了第一处理器，第二可执行代码被发往了第二处理器，因此源代码中的代码段对应的可执行代码被分配到匹配度较高的硬件来执行，因此执行效率会更高，实现了硬件加速。

请参阅图 10，本发明实施例中编译器包括：

获取单元 901，用于获取编译策略信息以及源代码；该编译策略信息指示第一代码类型与第一处理器匹配，第二代码类型与第二处理器匹配；

第一编译单元 902，用于根据该编译策略信息将该源代码中的代码段编译为与第一处理器匹配的第一可执行代码；

第一发送单元 903，用于将该第一可执行代码发往该第一处理器；

第一接收单元 904，用于接收第一处理器统计的第一可执行代码的第一执行信息；

第一处理单元 905，用于依据该第一执行信息确定该第一可执行代码对应的代码段是否与第二处理器匹配；

第二编译单元 906，用于该第一处理单元 905 依据该第一执行信息确定该第一可执行代码对应的代码段与第二处理器匹配，则将该第一可执行代码对应的代码段编译为第二可执行代码；

第二发送单元 907，用于将该第二可执行代码发往第二处理器。

可选的，本实施例还包括：

第二接收单元 908，用于接收来自该第二处理器执行该第二可执行代码的第二执行信息；

第二处理单元 909，用于若依据该第二执行信息确定该第二可执行代码对应的代码段与该第二处理器匹配度低于期望值，则在该第二处理器卸载该第二可执行代码，将该第二可执行代码对应的代码段对应的第一可执行代码发往该第一处理器。

本实施例中，将源代码中的代码段编译为与第一处理器匹配的第一可执行代码，当确定第一可执行代码对应的代码段与第二处理器匹配时，将第一可执行代码对应的代码段编译为第二可执行代码，通过第二处理器接收第二可执行代码，通过不同的硬件执行不同的代码类型，提高了执行效率。

此外，当第二可执行代码在第二处理器执行效率不高时，则在第二处理器上卸载该第二可执行代码，并对第二可执行代码对应的代码段重新编译，通过第一处理器来执行。

为了便于理解，下面以一实际的应用场景对本实施例中设备各单元间的交互进行描述：

调度管理器配置编译策略信息，调度管理器将该编译策略信息发送给获取单元 801，同时，获取单元 801 接收源代码。其中该编译策略信息指示第一代类型与第一处理器匹配，第二代类型与第二处理器匹配。代码类型包括：可加速代码以及普通代码。其中可加速代码可以为：CPU 指令，代码块，函数（模块）以及源文件。在获取编译策略信息以及源代码后，确定单元 802 根据该编译策略信息分析该源代码中的代码段，确定属于第一代类型的第一代代码段或属于第二代类型的第二代代码段。

编译单元 803 通过编译器中的编译单元将第一代代码段编译为第一可执行代码，发送单元 804 将该第一可执行代码通过调度管理器发往上述第一处理器。同理，编译单元 803 将第二代代码段编译为第二可执行代码，由于不同的执行逻辑（任务，进程，线程等）具有不同的优先级，因此存在一种可能，即上述第二代代码段对应进程的优先级高于正在该

第二处理器中执行的可执行代码对应的进程的优先级。当第二处理器中没有足够的资源来执行第二可执行代码时（也即是通过第一处理单元 805 说明第二处理器的繁忙度高于预设阈值），则将第二处理器中正在执行的可执行代码占用的资源释放出来（也即是停止正在该第二处理器中执行的可执行代码）；第二处理单元 806 将正在第二处理器中执行的可执行代码对应的代码段编译为与第一处理器匹配的可执行代码，发送给该第一处理器；这样第二处理器就有足够的资源来执行第二可执行代码，将该第二可执行代码发往上述第二处理器。

请参阅图 11，本发明实施例中编译器包括：

输入单元 1001、编译单元 1002 和输出单元 1003；

10 其中输入单元 1001，用于接收源代码、编译策略信息、硬件信息以及可加速代码描述特征库；

编译单元 1002，用于编译 CPU 可执行代码以及 FPGA 可执行代码；

输出单元 1003，用于将 CPU 可执行代码和 FPGA 可执行代码发送出去。

15 其中编译单元 1002 有三个核心的功能组件，可参考图 12，分别是：编译控制单元、CPU 编译单元以及 FPGA 编译单元。

编译策略信息和硬件信息可以通过外部程序输入，也可以由编译控制单元进行操作。

20 编译控制单元识别并执行编译策略信息，识别和处理可加速代码描述特征库，增减和修改规则，还可以根据可加速代码描述特征库、编译侧罗和硬件信息来检测源代码中是否有可加速的代码，然后根据检测结果生成中间源代码。

CPU 编译单元用来编译生成 CPU 可执行代码，FPGA 编译单元用来编译生成 FPGA 可执行代码。FPGA 编译单元可以直接根据源代码或中间源代码来编译，也可以将可加速代码或者中间源代码翻译转换成逻辑语言代码，再将逻辑语言代码编译成 FPGA 可执行代码。

25 需要说明的是，一个编译控制单元可以管理一个或多个 CPU 编译单元，一个编译控制单元可以管理一个或多个 FPGA 编译单元，一个 CPU 编译单元或 FPGA 编译单元可以受一个或多个编译控制单元管理。

本发明实施例还提供一种设备，请参阅图 13，本发明实施例中设备包括：

编译器和调度管理器，该编译器包括：

30 获取单元 1101、确定单元 1102、编译单元 1103 以及发送单元 1104；

获取单元 1101，用于获取编译策略信息以及源代码；该编译策略信息指示第一代

类型与第一处理器匹配，第二代码类型与第二处理器匹配；

确定单元 1102，用于根据该编译策略信息分析该源代码中的代码段，确定属于第一代码类型的第一代码段或属于第二代码类型的第二代码段；

5 编译单元 1103，用于将第一代码段编译为第一可执行代码，将第二代码段编译为第二可执行代码；

发送单元 1104，用于将该第一可执行代码和该第二可执行代码发送给调度管理器；

调度管理器，用于将该第一可执行代码发往该第一处理器，将该第二可执行代码发往该第二处理器。

可选的，调度管理器，还用于配置编译策略信息，将编译策略信息发送给编译器。

10 调度管理器，还用于若该第二处理器的繁忙度高于第二预定阈值，在该编译器停止正在该第二处理器中执行的可执行代码，将正在该第二处理器中执行的可执行代码对应的代码段编译为与第一处理器匹配的可执行代码后，接收来自该与第一处理器匹配的可执行代码，并发送给第一处理器。

进一步可选的，若第一处理器为 CPU，第二处理器为 FPGA，确定单元 1102，还用于调用统计指令集通过该源代码中的功能代码确定该源代码中的代码段的循环执行次数和/或 CPU 的执行时间；或者

调度管理器，还用于获取定时统计的代码执行统计报表，将该定时统计的代码执行统计报表发送给该编译器；

20 确定单元 1102，还用于依据该代码执行统计报表确定该源代码中的代码段的循环执行次数和/或 CPU 的执行时间；

确定单元 1102 用于确定属于第一代码类型的第一代码段或属于第二代码类型的第二代码段包括：若该循环执行次数和/或 CPU 的执行时间超过第一预定阈值，则确定该代码段属于第二代码段，否则确定该代码段属于第一代码段。

本发明实施例还提供一种设备，请参阅图 14，本发明实施例中设备包括：

25 编译器和调度管理器，该编译器包括：

获取单元 1201、编译单元 1202、发送单元 1203、接收单元 1204 以及处理单元 1205；

获取单元 1201，用于获取编译策略信息以及源代码；该编译策略信息指示第一代码类型与第一处理器匹配，第二代码类型与第二处理器匹配；

30 编译单元 1202，用于根据该编译策略信息将该源代码中的代码段编译为与第一处理器匹配的第一可执行代码；

发送单元 1203，用于将该第一可执行代码发送给第一处理器；

接收单元 1204, 用于接收第一处理器统计的第一可执行代码的第一执行信息;

处理单元 1205, 用于依据该第一执行信息确定该第一可执行代码对应的代码段是否与第二处理器匹配;

5 编译单元 1202, 还用于该处理单元 1205 依据该第一执行信息确定第一可执行代码对应的代码段与第二处理器匹配, 则将该第一可执行代码对应的代码段编译为第二可执行代码;

发送单元 1203, 还用于将该第二可执行代码发送给调度管理器;

调度管理器, 用于将第二可执行代码发往该第二处理器。

10 可选的, 调度管理器, 还用于接收来自该第二处理器执行该第二可执行代码的第二执行信息, 依据该第二执行信息确定该第二可执行代码对应的代码段与该第二处理器匹配度低于期望值, 则在该第二处理器卸载该第二可执行代码; 将该第二可执行代码对应的代码段对应的第一可执行代码发往该第一处理器。

本发明实施例还提供一种服务器, 请参阅图 15, 本发明实施例中服务器包括:

15 该服务器 1300 可因配置或性能不同而产生比较大的差异, 可以包括编译器 1301、调度管理器 1302、FPGA1303、一个或一个以上 CPU1304 (例如, 一个或一个以上处理器) 和存储器 1305, 一个或一个以上存储应用程序 1306 或数据 1307 的存储介质 1308 (例如一个或一个以上海量存储设备)。其中, 存储器 1305 和存储介质 1308 可以是短暂存储或持久存储。存储在存储介质 1308 的程序可以包括一个或一个以上模块 (图示没标出), 每个模块可以包括对服务器中的一系列指令操作。更进一步地, 中央处理器 20 1304 可以设置为与存储介质 1308 通信, 在服务器 1301 上执行存储介质 1305 中的一系列指令操作。

服务器 1300 还可以包括一个或一个以上电源 1309, 一个或一个以上有线或无线网络接口 1310, 一个或一个以上输入输出接口 1311, 和/或, 一个或一个以上操作系统 1312, 例如 Windows Server™, Mac OS X™, Unix™, Linux™, FreeBSD™ 等等。

25 值得注意的是, 上述设备和编译器实施例中, 所包括的各个单元只是按照功能逻辑进行划分的, 但并不局限于上述的划分, 只要能够实现相应的功能即可; 另外, 各功能单元的具体名称也只是为了便于相互区分, 并不用于限制本发明的保护范围。

30 另外, 本领域普通技术人员可以理解实现上述各方法实施例中的全部或部分步骤是可以通程序来指令相关的硬件完成, 相应的程序可以存储于一种计算机可读存储介质中, 上述提到的存储介质可以是只读存储器, 磁盘或光盘等。

以上仅为本发明较佳的具体实施方式, 但本发明的保护范围并不局限于此, 任何熟悉本技术领域的技术人员在本发明实施例揭露的技术范围内, 可轻易想到的变化或替换, 都应涵盖在本发明的保护范围之内。因此, 本发明的保护范围应该以权利要求的保护范围为准。

## 权利要求

1、一种硬件加速方法，其特征在于，包括：

编译器获取编译策略信息以及源代码；所述编译策略信息指示第一代类型与第一处理器匹配，第二代类型与第二处理器匹配；

5 所述编译器根据所述编译策略信息分析所述源代码中的代码段，确定属于第一代类型的第一代代码段或属于第二代类型的第二代代码段；

所述编译器将第一代代码段编译为第一可执行代码，将所述第一可执行代码发往所述第一处理器；将第二代代码段编译为第二可执行代码，将所述第二可执行代码发往所述第二处理器。

10 2、根据权利要求 1 所述方法，其特征在于，所述将所述第二可执行代码发往所述第二处理器包括：

所述编译器通过调度管理器将所述第二可执行代码发送给所述第二处理器；或者，

所述编译器在所述第二可执行代码中添加调用第二处理器的指示信息，通过调度管理器将所述第二可执行代码发送给所述第一处理器，使所述第一处理器获取到所述指示信息后将所述第二可执行代码发往所述第二处理器。

15 3、根据权利要求 1 所述方法，其特征在于，所述第一处理器为 CPU，第二处理器为 FPGA；所述编译器根据所述编译策略信息分析所述源代码中的代码段，确定属于第一代类型的第一代码段或属于第二代类型的第二代代码段包括：

20 所述编译器统计所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间，若统计得到的参数超过第一预定阈值，则确定该代码段属于第二代代码段，否则确定该代码段属于第一代代码段；或者，

所述编译器将所述源代码中的代码段与所述第二代代码段的特征规则进行比较，若与所述第二代代码段的特征规则匹配，则确定该代码段为第二代代码段，否则，确定该代码段为第一代代码段；或者，

25 所述编译器读取所述源代码中的加速标识信息，依据所述加速标识信息的指示确定所述源代码中的代码属于第一代代码段或第二代代码段。

4、根据权利要求 3 所述方法，其特征在于，所述编译器统计所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间包括：

30 所述编译器调用统计指令集通过所述源代码中的功能代码确定所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间；或者，所述编译器获取定时统计的代码执行统计报表确定所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间。

5、根据权利要求 1 至 4 任意一项所述方法，其特征在于，若所述第二代码段对应进程的优先级高于正在所述第二处理器中执行的可执行代码对应的进程的优先级，在将所述第二可执行代码发往所述第二处理器之前，所述方法还包括：

若所述第二处理器的繁忙度高于第二预定阈值，则停止正在所述第二处理器中执行的可执行代码；将正在所述第二处理器中执行的可执行代码对应的代码段编译为与第一处理器匹配的可执行代码，发送给所述第一处理器。

6、根据权利要求 1 至 4 任意一项所述方法，其特征在于，若第一代码段属于主代码段，所述第二代码段从属所述主代码段，则所述方法还包括：

在所述第一可执行代码以及所述第二可执行代码中添加交互指令，使所述第二处理器向所述第一处理器返回所述第二可执行代码的执行结果。

7、一种硬件加速方法，其特征在于，包括：

编译器获取编译策略信息以及源代码；所述编译策略信息指示第一代码类型与第一处理器匹配，第二代码类型与第二处理器匹配；

所述编译器根据所述编译策略信息将所述源代码中的代码段编译为与第一处理器匹配的第一可执行代码，将所述第一可执行代码发往所述第一处理器；

若接收到所述第一处理器统计的所述第一可执行代码的第一执行信息，并依据所述第一执行信息确定所述第一可执行代码对应的代码段与第二处理器匹配，则将所述第一可执行代码对应的代码段编译为第二可执行代码，将所述第二可执行代码发往第二处理器。

8、根据权利要求 7 所述方法，其特征在于，在将所述第二可执行代码发往第二处理器之后，所述方法还包括：

接收来自所述第二处理器执行所述第二可执行代码的第二执行信息，若依据所述第二执行信息确定所述第二可执行代码对应的代码段与所述第二处理器匹配度低于期望值，则在所述第二处理器卸载所述第二可执行代码，将所述第二可执行代码对应的代码段对应的第一可执行代码发往所述第一处理器。

9、一种编译器，其特征在于，包括：

获取单元，用于获取编译策略信息以及源代码；所述编译策略信息指示第一代码类型与第一处理器匹配，第二代码类型与第二处理器匹配；

确定单元，用于根据所述编译策略信息分析所述源代码中的代码段，确定属于第一代码类型的第一代码段或属于第二代码类型的第二代码段；

编译单元，用于将第一代码段编译为第一可执行代码，将第二代码段编译为第二可

执行代码；

发送单元，用于将所述第一可执行代码发往所述第一处理器，将所述第二可执行代码发往所述第二处理器。

10、根据权利要求 9 所述的编译器，其特征在于，所述发送单元用于通过调度管理器将所述第二可执行代码发送给所述第二处理器；或者，

所述编译单元还用于在所述第二可执行代码中添加调用第二处理器的指示信息；

所述发送单元还用于将所述第二可执行代码发送给调度管理器，通过调度管理器将所述第二可执行代码发送给所述第一处理器，使所述第一处理器获取到所述指示信息后将所述第二可执行代码发往所述第二处理器。

10 11、根据权利要求 9 所述的编译器，其特征在于，所述第一处理器为 CPU，第二处理器为 FPGA；所述确定单元用于统计所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间，若统计得到的参数超过第一预定阈值，则确定该代码段属于第二代码段，否则确定该代码段属于第一代码段；或者，

15 所述确定单元用于将所述源代码中的代码段与所述第二代码段的特征规则进行比较，若与所述第二代码段的特征规则匹配，则确定该代码段为第二代码段，否则，确定该代码段为第一代码段；或者，

所述确定单元用于读取所述源代码中的加速标识信息，依据所述加速标识信息的指示确定所述源代码中的代码属于第一代码段或第二代码段。

20 12、根据权利要求 11 所述的编译器，其特征在于，所述确定单元用于调用统计指令集通过所述源代码中的功能代码确定所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间；或者，

所述确定单元用于获取定时统计的代码执行统计报表确定所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间。

25 13、根据权利要求 9 至 12 任意一项所述的编译器，其特征在于，若所述第二代码段对应进程的优先级高于正在所述第二处理器中执行的可执行代码对应的进程的优先级，所述设备还包括：

第一处理单元，用于若所述第二处理器的繁忙度高于第二预定阈值，则停止正在所述第二处理器中执行的可执行代码；

30 第二处理单元，用于将正在所述第二处理器中执行的可执行代码对应的代码段编译为与第一处理器匹配的可执行代码，发送给所述第一处理器。

14、根据权利要求 9 至 12 任意一项所述的编译器，其特征在于，若第一代码段属



于主代码段，所述第二代码段从属所述主代码段，所述编译器还包括：

第三处理单元，用于在所述第一可执行代码以及所述第二可执行代码中添加交互指令，使所述第二处理器向所述第一处理器返回所述第二可执行代码的执行结果。

15、一种编译器，其特征在于，包括：

5 获取单元，用于获取编译策略信息以及源代码；所述编译策略信息指示第一代类型与第一处理器匹配，第二代类型与第二处理器匹配；

第一编译单元，用于根据所述编译策略信息将所述源代码中的代码段编译为与第一处理器匹配的第一可执行代码；

第一发送单元，用于将所述第一可执行代码发往所述第一处理器；

10 第一接收单元，用于接收所述第一处理器统计的所述第一可执行代码的第一执行信息；

第一处理单元，用于依据所述第一执行信息确定所述第一可执行代码对应的代码段是否与第二处理器匹配；

15 第二编译单元，用于所述第一处理单元依据所述第一执行信息确定所述第一可执行代码对应的代码段与第二处理器匹配，则将所述第一可执行代码对应的代码段编译为第二可执行代码；

第二发送单元，用于将所述第二可执行代码发往第二处理器。

16、根据权利要求 15 所述的编译器，其特征在于，所述编译器还包括：

20 第二接收单元，用于接收来自所述第二处理器执行所述第二可执行代码的第二执行信息；

第二处理单元，用于若依据所述第二执行信息确定所述第二可执行代码对应的代码段与所述第二处理器匹配度低于期望值，则在所述第二处理器卸载所述第二可执行代码，将所述第二可执行代码对应的代码段对应的第一可执行代码发往所述第一处理器。

17、一种设备，其特征在于，包括：

25 编译器和调度管理器，所述编译器包括：

获取单元、确定单元、编译单元以及发送单元；

所述获取单元，用于获取编译策略信息以及源代码；所述编译策略信息指示第一代类型与第一处理器匹配，第二代类型与第二处理器匹配；

30 所述确定单元，用于根据所述编译策略信息分析所述源代码中的代码段，确定属于第一代类型的第一代代码段或属于第二代类型的第二代代码段；

所述编译单元，用于将第一代代码段编译为第一可执行代码，将第二代代码段编译为第

二可执行代码；

所述发送单元，用于将所述第一可执行代码和所述第二可执行代码发送给调度管理器；

5 所述调度管理器，用于将所述第一可执行代码发往所述第一处理器，将所述第二可执行代码发往所述第二处理器。

18、根据权利要求 17 所述的设备，其特征在于，

所述调度管理器，还用于配置编译策略信息，将所述编译策略信息发送给所述编译器。

10 19、根据权利要求 17 所述的设备，其特征在于，若所述第一处理器为 CPU，所述第二处理器为 FPGA，

所述确定单元，还用于调用统计指令集通过所述源代码中的功能代码确定所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间；或者

所述调度管理器，还用于获取定时统计的代码执行统计报表，将所述定时统计的代码执行统计报表发送给所述编译器；

15 所述确定单元，还用于依据所述代码执行统计报表确定所述源代码中的代码段的循环执行次数和/或 CPU 的执行时间；

所述确定单元用于确定属于第一代码类型的第一代码段或属于第二代码类型的第二代码段包括：若所述循环执行次数和/或 CPU 的执行时间超过第一预定阈值，则确定该代码段属于第二代码段，否则确定该代码段属于第一代码段。

20 20、根据权利要求 17 所述的设备，其特征在于，

所述调度管理器，还用于若所述第二处理器的繁忙度高于第二预定阈值，在所述编译器停止正在所述第二处理器中执行的可执行代码，将正在所述第二处理器中执行的可执行代码对应的代码段编译为与第一处理器匹配的可执行代码后，接收来自所述与第一处理器匹配的可执行代码，并发送给第一处理器。

25

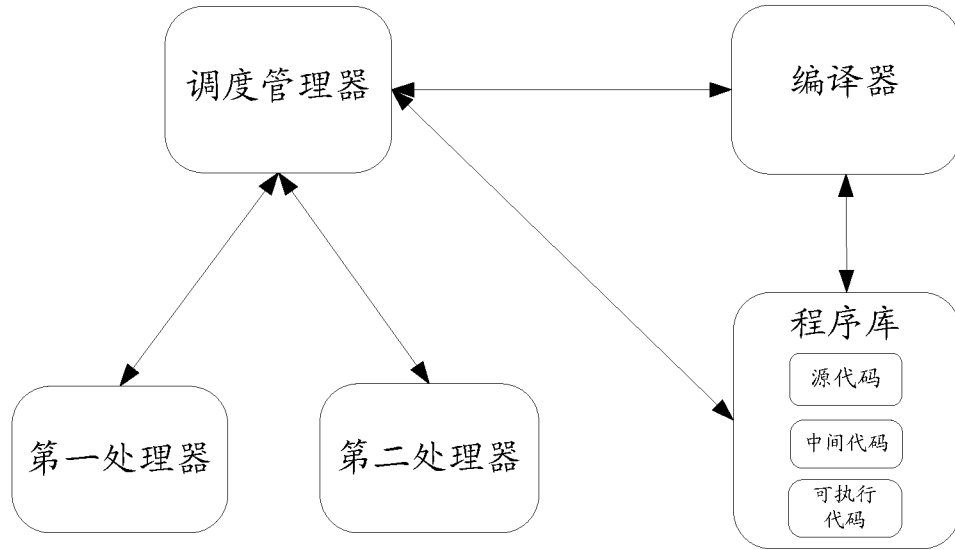


图 1

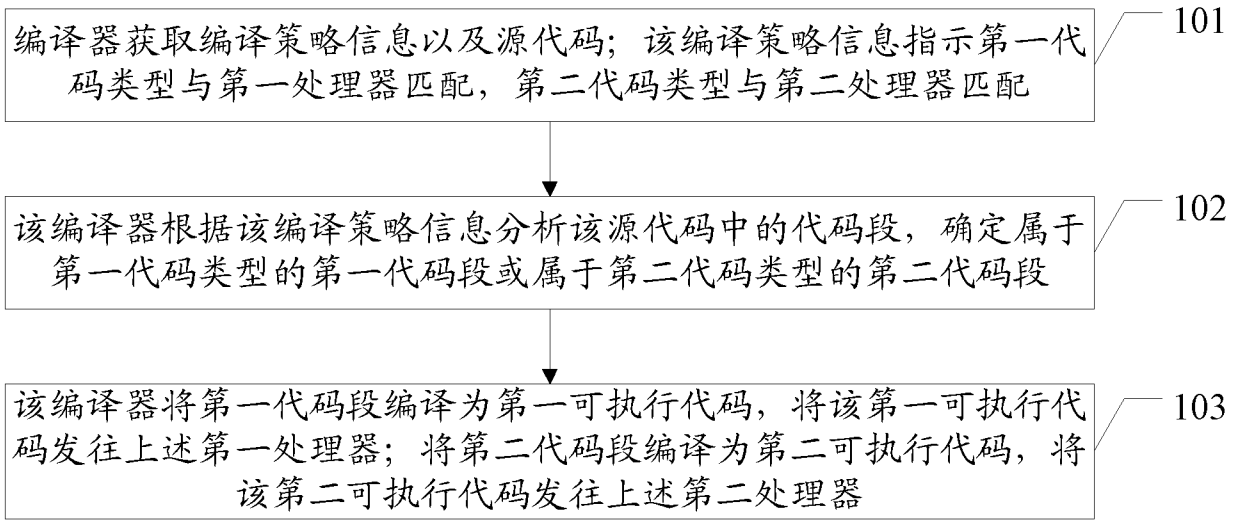


图 2

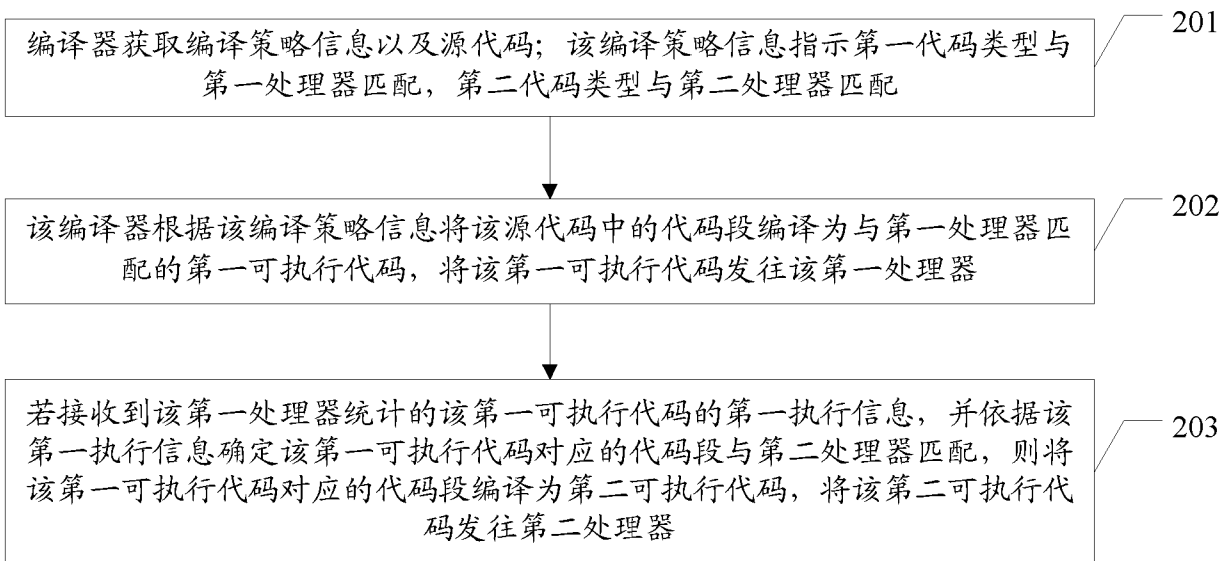


图 3

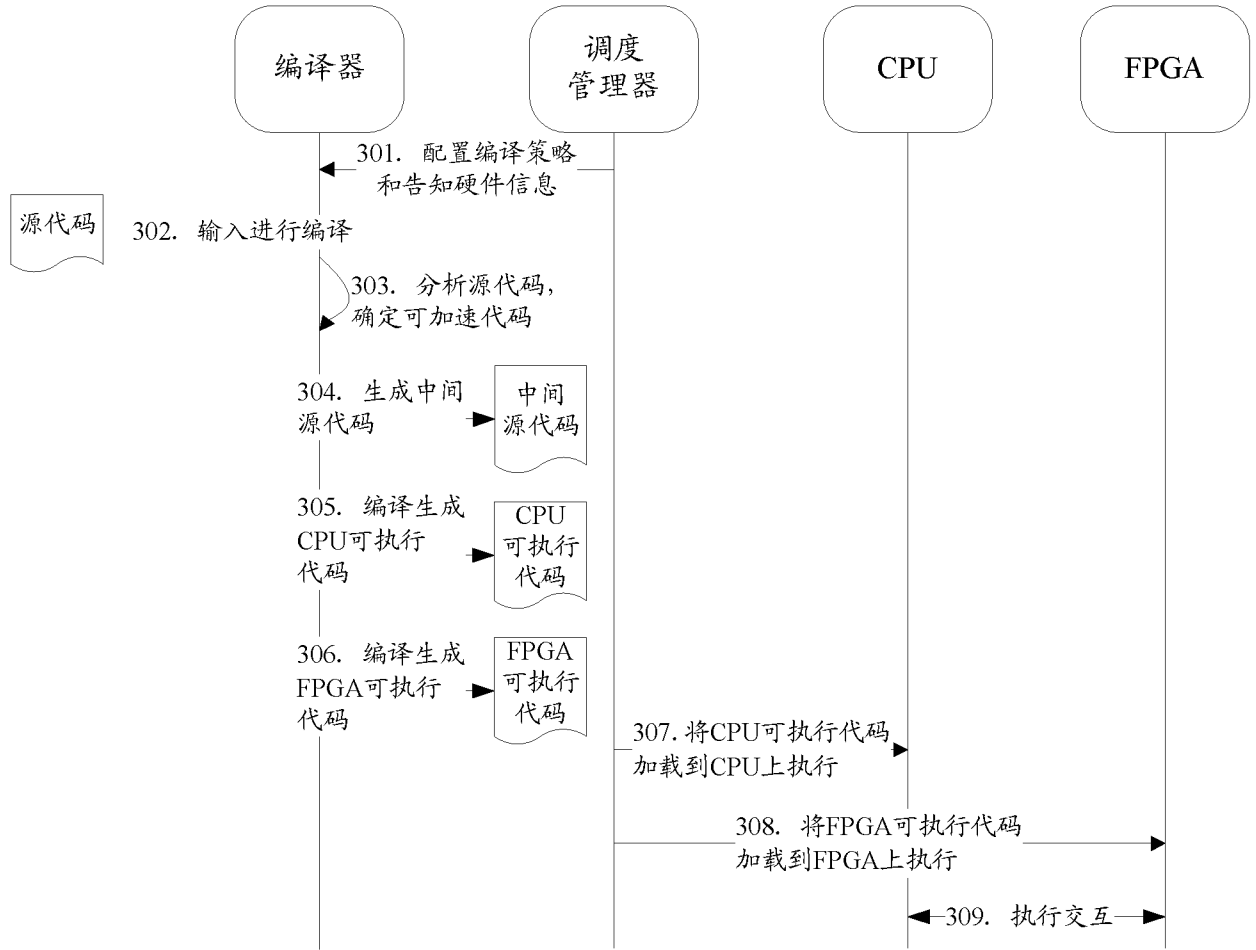


图 4

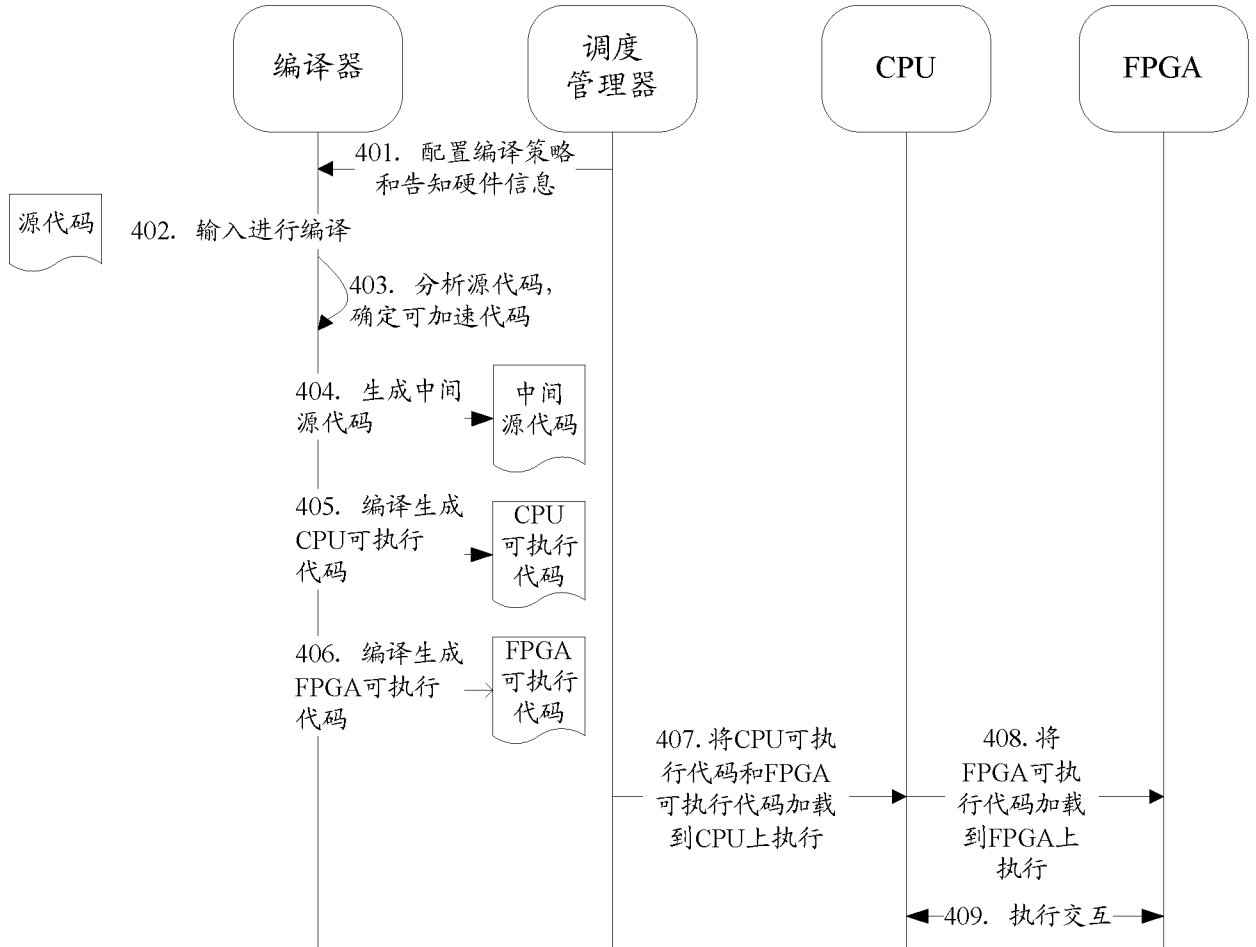


图 5

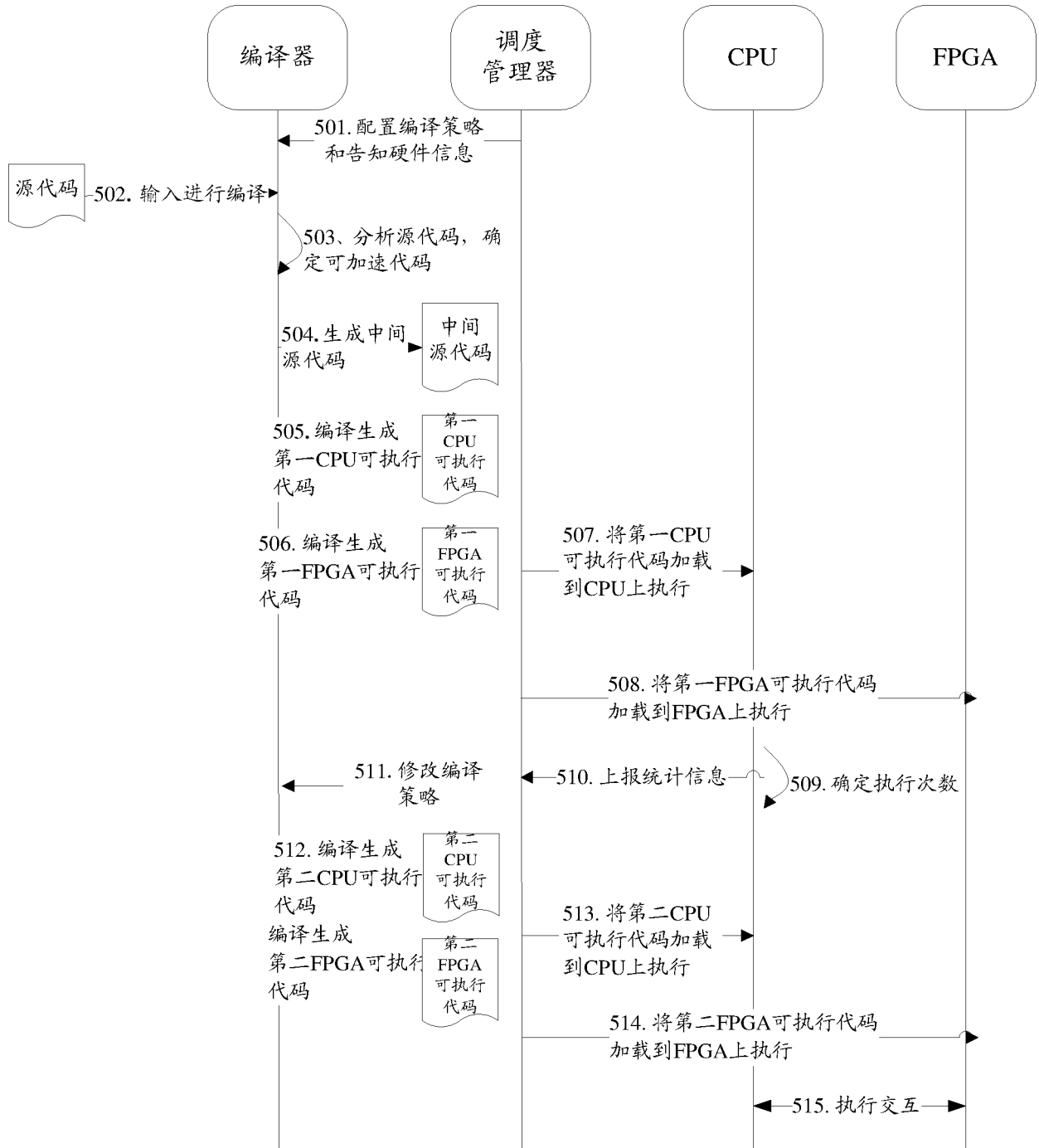


图 6

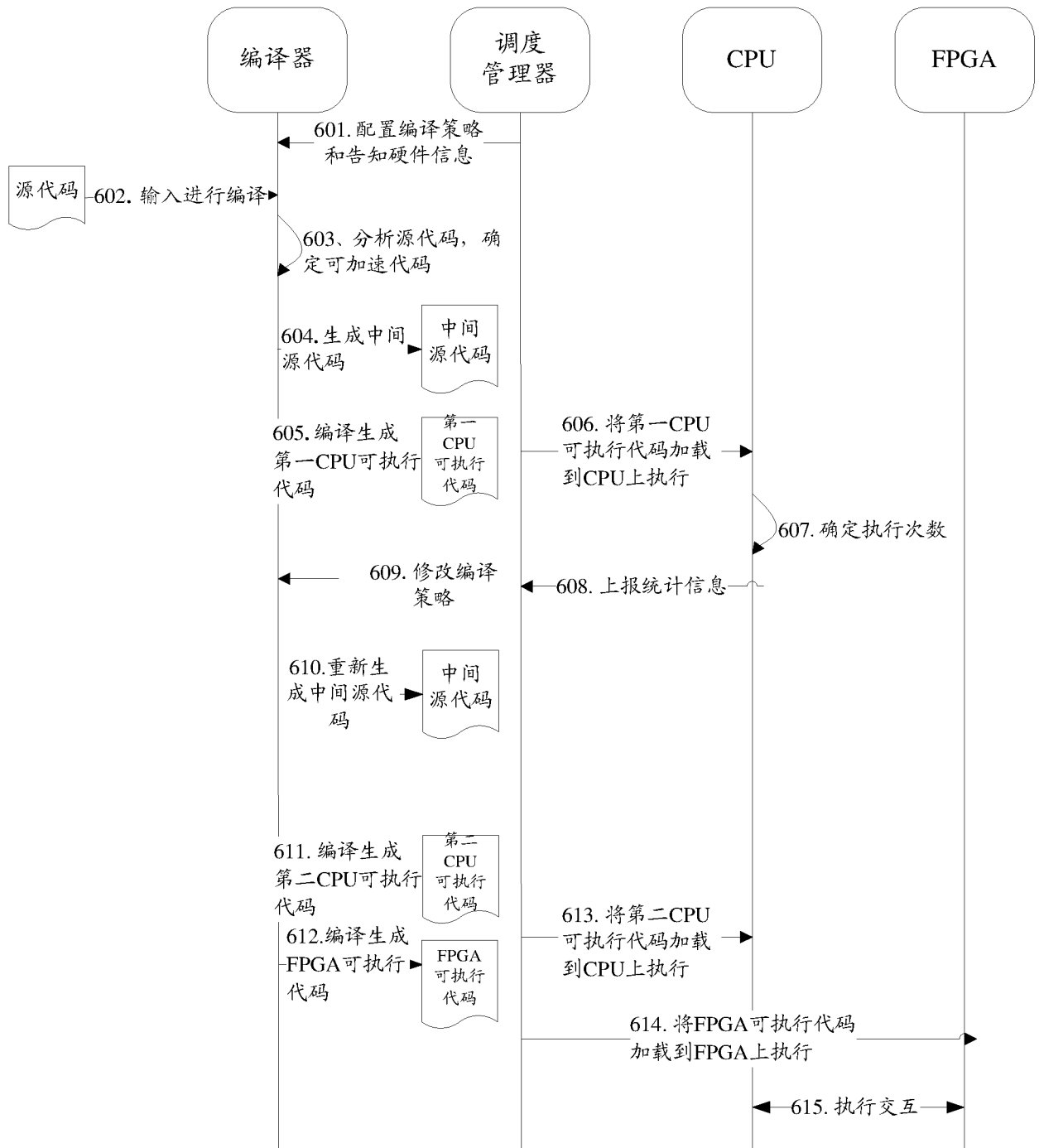


图 7

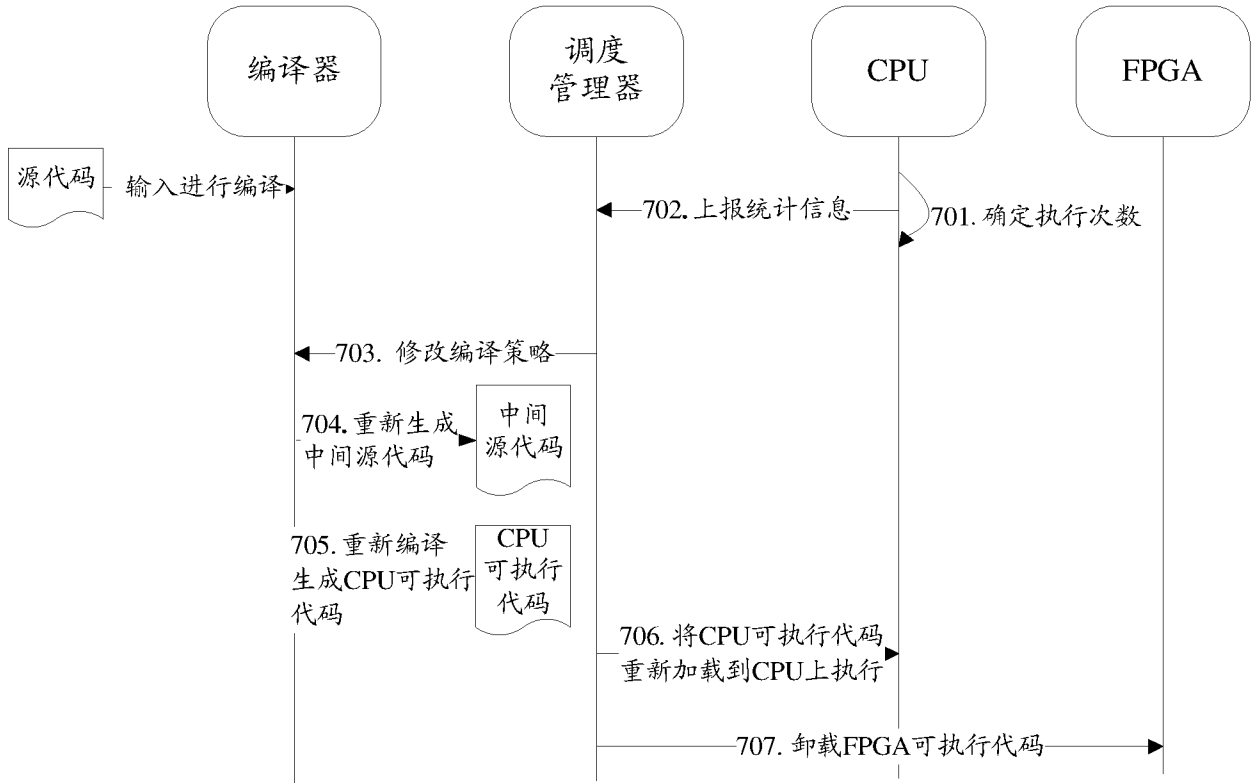


图 8

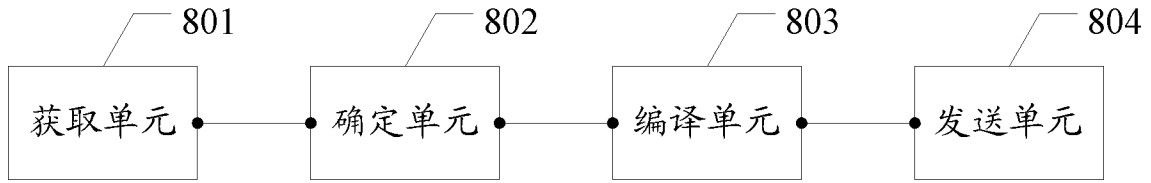


图 9

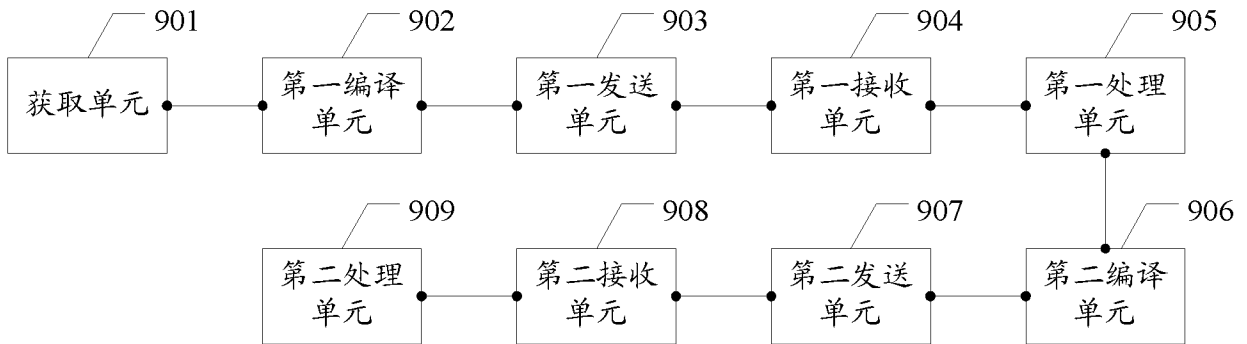


图 10

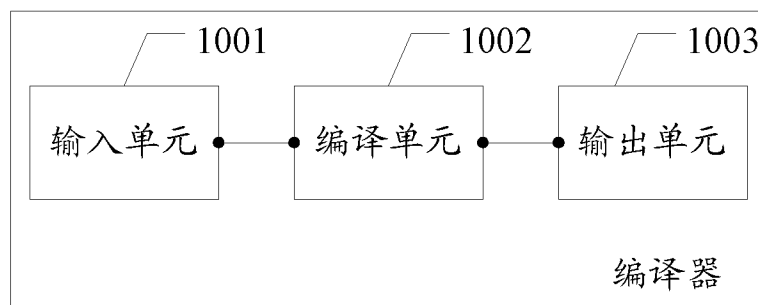


图 11



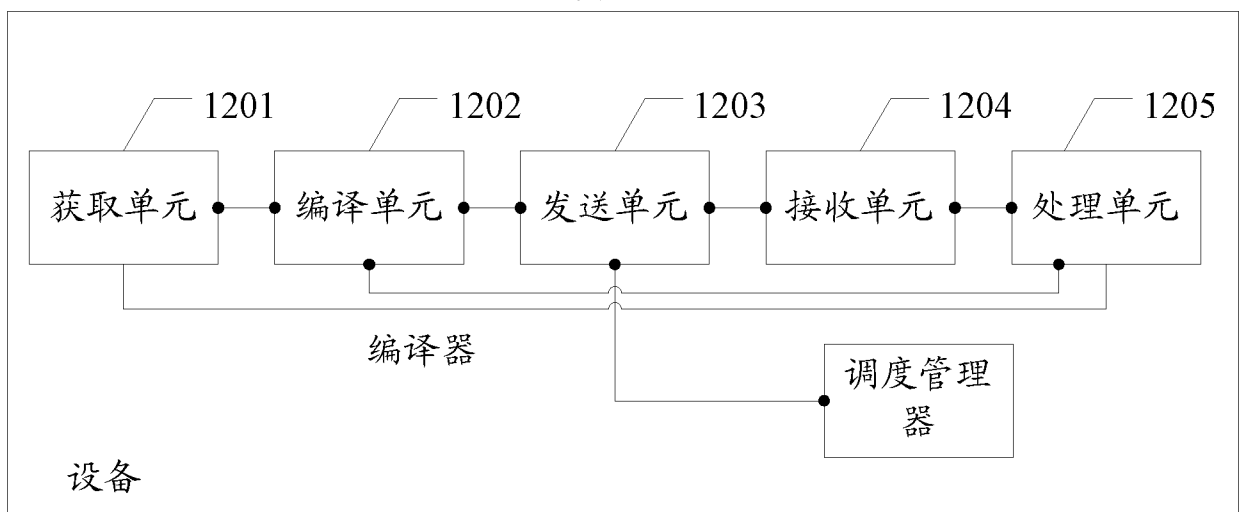
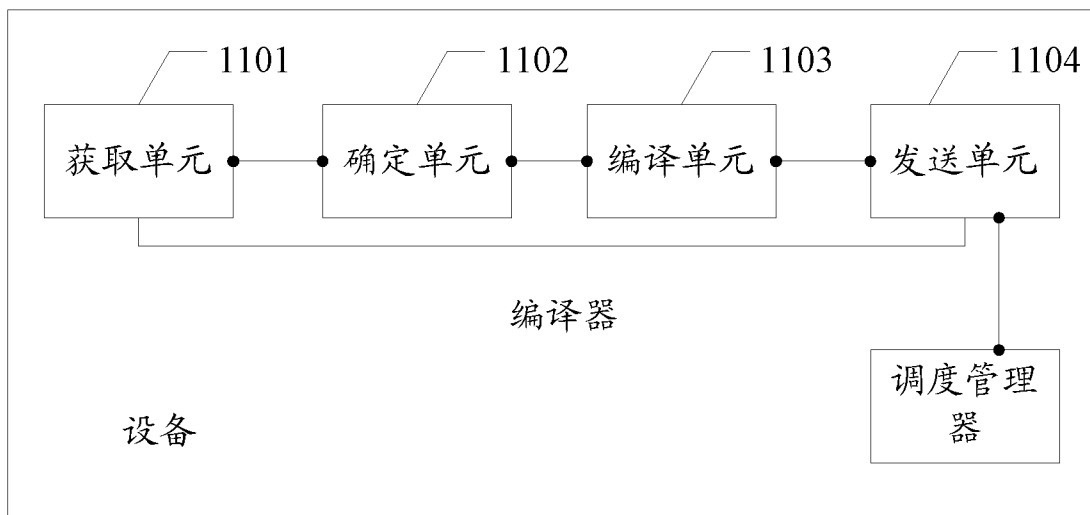
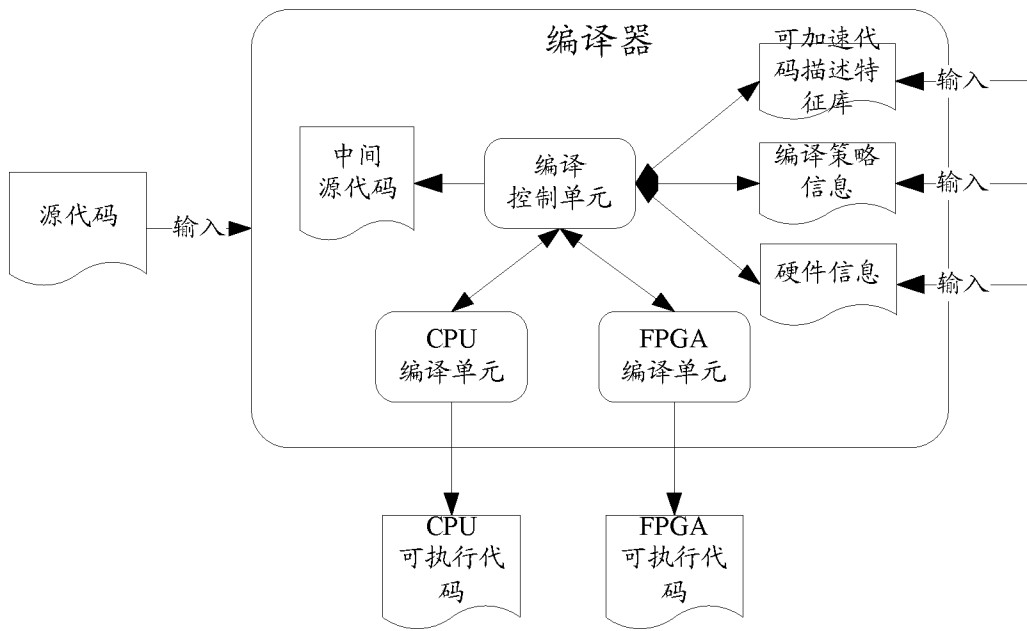


图 14

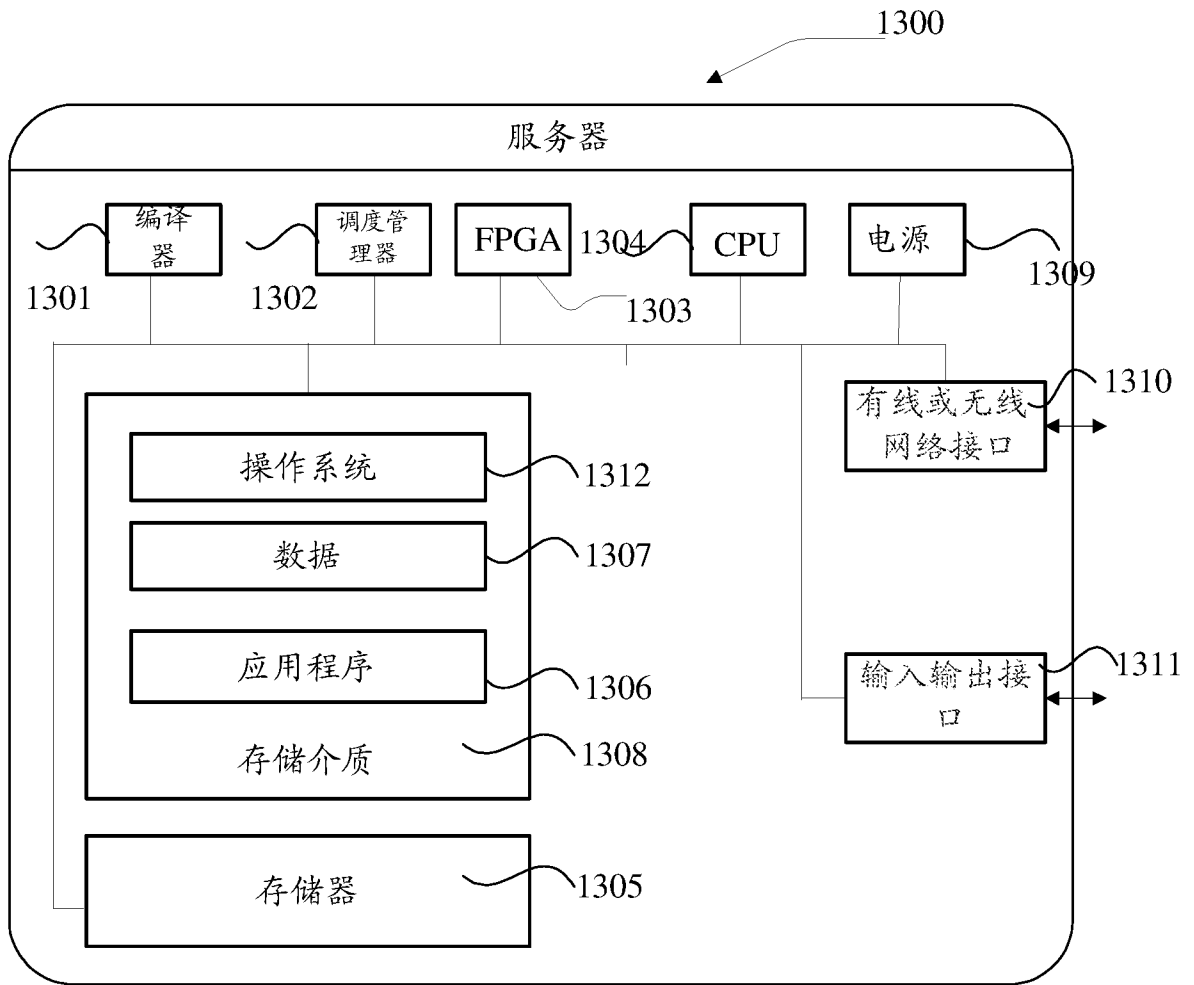


图 15

# INTERNATIONAL SEARCH REPORT

International application No.

**PCT/CN2016/087051**

## A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/45 (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)

CNXTX; CNABS; CNKI; DWPI; SIPOABS: overhead; processor, multiprocessor, heterogeneous, compile, type, send, code, match, accelerate, cost

## C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2015052331 A1 (QUALCOMM INC.), 19 February 2015 (19.02.2015), description, paragraphs [0018]-[0066], and figures 1-5	1-20
A	US 6292938 B1 (INTERNATIONAL BUSINESS MACHINES CORPORATION), 18 September 2001 (18.09.2001), the whole document	1-20
A	CN 1975753 A (HE, Bo et al.), 06 June 2007 (06.06.2007), the whole document	1-20

Further documents are listed in the continuation of Box C.

See patent family annex.

<p>* Special categories of cited documents:</p> <p>“A” document defining the general state of the art which is not considered to be of particular relevance</p> <p>“E” earlier application or patent but published on or after the international filing date</p> <p>“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)</p> <p>“O” document referring to an oral disclosure, use, exhibition or other means</p> <p>“P” document published prior to the international filing date but later than the priority date claimed</p>	<p>“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</p> <p>“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</p> <p>“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</p> <p>“&amp;” document member of the same patent family</p>
---	---

Date of the actual completion of the international search

19 September 2016 (19.09.2016)

Date of mailing of the international search report

**29 September 2016 (29.09.2016)**

Name and mailing address of the ISA/CN:  
 State Intellectual Property Office of the P. R. China  
 No. 6, Xitucheng Road, Jimenqiao  
 Haidian District, Beijing 100088, China  
 Facsimile No.: (86-10) 62019451

Authorized officer

**YANG, Peng**

Telephone No.: (86-10) **62088420**

**INTERNATIONAL SEARCH REPORT**  
Information on patent family members

International application No.

**PCT/CN2016/087051**

Patent Documents referred in the Report	Publication Date	Patent Family	Publication Date
US 2015052331 A1	19 February 2015	CN 105474172 A	06 April 2016
		US 9201659 B2	01 December 2015
		EP 3036624 A1	29 June 2016
		WO 2015026554 A1	26 February 2015
		IN 201627000448 A	22 July 2016
US 6292938 B1	18 September 2001	KR 100334995 B1	02 May 2002
		KR 20000052396 A	25 August 2000
CN 1975753 A	06 June 2007	CN 100446023 C	24 December 2008

<p>A. 主题的分类</p> <p>G06F 9/45 (2006.01) i</p> <p>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</p>														
<p>B. 检索领域</p> <p>检索的最低限度文献 (标明分类系统和分类号)</p> <p>G06F</p> <p>包含在检索领域中的除最低限度文献以外的检索文献</p> <p>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))</p> <p>CNXTX; CNABS; CNKI; DWPI; SIPOABS: 处理器, 多处理器, 异构, 编译, 类型, 发送, 代码, 匹配, 加速, 开销; processor, multiprocessor, heterogeneous, compile, type, send, code, match, accelerate, cost</p>														
<p>C. 相关文件</p> <table border="1"> <thead> <tr> <th>类型*</th> <th>引用文件, 必要时, 指明相关段落</th> <th>相关的权利要求</th> </tr> </thead> <tbody> <tr> <td>X</td> <td>US 2015052331 A1 (高通股份有限公司) 2015年 2月 19日 (2015 - 02 - 19) 说明书第[0018]-[0066]段, 图1-5</td> <td>1-20</td> </tr> <tr> <td>A</td> <td>US 6292938 B1 (国际商业机器公司) 2001年 9月 18日 (2001 - 09 - 18) 全文</td> <td>1-20</td> </tr> <tr> <td>A</td> <td>CN 1975753 A (何波 等) 2007年 6月 6日 (2007 - 06 - 06) 全文</td> <td>1-20</td> </tr> </tbody> </table>			类型*	引用文件, 必要时, 指明相关段落	相关的权利要求	X	US 2015052331 A1 (高通股份有限公司) 2015年 2月 19日 (2015 - 02 - 19) 说明书第[0018]-[0066]段, 图1-5	1-20	A	US 6292938 B1 (国际商业机器公司) 2001年 9月 18日 (2001 - 09 - 18) 全文	1-20	A	CN 1975753 A (何波 等) 2007年 6月 6日 (2007 - 06 - 06) 全文	1-20
类型*	引用文件, 必要时, 指明相关段落	相关的权利要求												
X	US 2015052331 A1 (高通股份有限公司) 2015年 2月 19日 (2015 - 02 - 19) 说明书第[0018]-[0066]段, 图1-5	1-20												
A	US 6292938 B1 (国际商业机器公司) 2001年 9月 18日 (2001 - 09 - 18) 全文	1-20												
A	CN 1975753 A (何波 等) 2007年 6月 6日 (2007 - 06 - 06) 全文	1-20												
<p><input type="checkbox"/> 其余文件在C栏的续页中列出。</p> <p><input checked="" type="checkbox"/> 见同族专利附件。</p>														
<p>* 引用文件的具体类型:</p> <p>“A” 认为不特别相关的表示了现有技术一般状态的文件</p> <p>“E” 在国际申请日的当天或之后公布的在先申请或专利</p> <p>“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)</p> <p>“O” 涉及口头公开、使用、展览或其他方式公开的文件</p> <p>“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</p> <p>“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</p> <p>“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</p> <p>“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</p> <p>“&amp;” 同族专利的文件</p>														
<p>国际检索实际完成的日期</p> <p>2016年 9月 19日</p>		<p>国际检索报告邮寄日期</p> <p>2016年 9月 29日</p>												
<p>ISA/CN的名称和邮寄地址</p> <p>中华人民共和国国家知识产权局 (ISA/CN) 中国北京市海淀区蓟门桥西土城路6号 100088</p> <p>传真号 (86-10) 62019451</p>		<p>授权官员</p> <p>杨鹏</p> <p>电话号码 (86-10) 62088420</p>												

国际检索报告  
关于同族专利的信息

国际申请号

PCT/CN2016/087051

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)
US	2015052331	A1	2015年 2月 19日	CN	105474172	A	2016年 4月 6日
				US	9201659	B2	2015年 12月 1日
				EP	3036624	A1	2016年 6月 29日
				WO	2015026554	A1	2015年 2月 26日
				IN	201627000448	A	2016年 7月 22日
US	6292938	B1	2001年 9月 18日	KR	100334995	B1	2002年 5月 2日
				KR	20000052396	A	2000年 8月 25日
CN	1975753	A	2007年 6月 6日	CN	100446023	C	2008年 12月 24日