(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 106325819 B
(45)授权公告日 2019.08.02

(21)申请号 201510336409.5

(22)申请日 2015.06.17

(65)同一申请的已公布的文献号
申请公布号 CN 106325819 A

(43)申请公布日 2017.01.11

(73)专利权人 华为技术有限公司
地址 518129 广东省深圳市龙岗区坂田华为总部办公楼
专利权人 中国科学院计算技术研究所

(72)发明人 高云伟 林鑫龙 詹剑锋

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

(56)对比文件
CN 101901149 A,2010.12.01,
CN 101901149 A,2010.12.01,
CN 101387969 A,2009.03.18,
CN 103294540 A,2013.09.11,
US 2009/0319662 A1,2009.12.24,
CN 102193788 A,2011.09.21,
CN 104050010 A,2014.09.17,
CN 104572307 A,2015.04.29,

审查员 刘畅

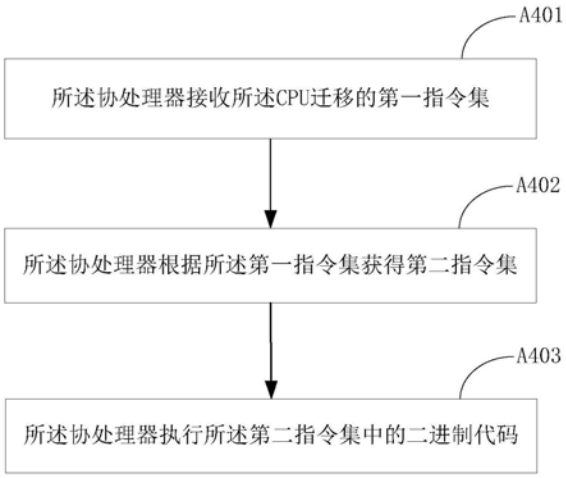
权利要求书4页 说明书27页 附图7页

(54)发明名称

计算机指令处理方法、协处理器和系统

(57)摘要

本发明实施例公开了一种计算机指令处理方法、协处理器和系统;所述计算机指令处理方法包括:协处理器接收中央处理器CPU迁移的第一指令集,根据适于CPU执行的第一指令集获取在协处理器执行的第二指令集,执行所述第二指令集中的二进制代码。这样,由协处理器执行第二指令集代替由CPU执行第一指令集,减小CPU的负荷,提高协处理器的使用率。




CN 106325819 B

1.一种计算机指令处理方法,应用于处理器系统,所述处理器系统包括协处理器和中央处理器CPU,其特征在于,所述CPU上运行第一操作系统,所述协处理器上运行第二操作系统;

所述方法包括:

所述协处理器接收所述CPU迁移的第一指令集,所述第一指令集用于指示所述CPU在所述第一操作系统中执行计算机操作,所述第一指令集为适用于所述第一操作系统的二进制代码的集合;

所述协处理器根据所述第一指令集获得第二指令集,其中,所述第二指令集中的二进制代码用于指示所述协处理器在所述第二操作系统中执行所述计算机操作;

所述协处理器执行所述第二指令集中的二进制代码;

其中,所述协处理器根据所述第一指令集获得第二指令集还包括:

若所述第一指令集中的第三二进制代码的操作码在翻译表中未被匹配到,并且所述第三二进制代码为操作码属于第三部分子指令集的二进制代码,则所述协处理器将所述第三二进制代码做为所述第二指令集中的二进制代码,所述协处理器的指令集不包含所述第三部分子指令集,所述翻译表包含相同的计算机指令分别编译生成的在所述第一操作系统和所述第二操作系统中不同的操作码之间的对应关系。

2.根据权利要求1所述的方法,其特征在于,所述协处理器根据所述第一指令集获得第二指令集包括:

所述协处理器在预先设置的所述翻译表中匹配所述第一指令集中的二进制代码的操作码,若所述第一指令集中的第一二进制代码的操作码在所述翻译表中被匹配到,则根据所述翻译表中所述第一二进制代码的操作码对应的匹配项,将所述第一二进制代码的操作码翻译为第二二进制代码的操作码,获得所述第二二进制代码,所述协处理器根据获得的至少一条所述第二二进制代码获得所述第二指令集,其中,所述翻译表包含相同的计算机指令分别编译生成的在所述第一操作系统和所述第二操作系统中不同的操作码之间的对应关系,所述第二二进制代码为适用于所述第二操作系统的二进制代码。

3.根据权利要求1所述的方法,其特征在于,在所述协处理器执行所述第二指令集中的二进制代码之前,所述方法还包括:

所述协处理器将所述第二指令集包含的二进制代码中所述CPU的寄存器地址转换为所述协处理器的寄存器地址。

4.根据权利要求1所述的方法,其特征在于,所述第一指令集是所述CPU在所述CPU的CPU使用率大于第一阈值时向所述协处理器迁移的。

5.根据权利要求4所述的方法,其特征在于,所述协处理器接收所述CPU迁移的第一指令集,包括:

所述协处理器接收所述CPU发送的要迁移的所述第一指令集的地址,所述第一指令集的地址是指所述第一指令集在所述CPU的内存中存储的地址,其中,所述第一指令集的地址由所述CPU在所述CPU的内存使用率小于或等于第二阈值时向所述协处理器发送;

所述协处理器基于所述第一指令集的地址访问所述CPU的内存来获取所述第一指令集。

6.根据权利要求1所述的方法,其特征在于,所述第一指令集由所述CPU在所述CPU的内

存使用率大于第二阈值时向所述协处理器发送。

7.根据权利要求1所述的方法，其特征在于，所述协处理器执行所述第二指令集中的二进制代码，包括：

所述协处理器依次执行所述第二指令集中的二进制代码；

如果检测到执行所述第二指令集时出现二进制代码识别异常，则确定触发异常的第四二进制代码；

将所述第四二进制代码转换为中间代码，再将所述中间代码转换为适用于所述第二操作系统的第五二进制代码；

执行所述第五二进制代码，并继续执行所述第二指令集中所述第四二进制代码之后的二进制代码。

8.根据权利要求7所述的方法，其特征在于，所述将所述第四二进制代码转换为中间代码之前，还包括：

向所述CPU发送指令集回迁请求；

接收所述CPU发送的拒绝回迁指令。

9.根据权利要求1至6任一项所述的方法，其特征在于，所述协处理器执行所述第二指令集中的二进制代码，包括：

所述协处理器依次执行所述第二指令集中的二进制代码；

如果检测到执行所述第二指令集时出现二进制代码识别异常，则确定触发异常的第六二进制代码；

根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集，并向所述CPU迁移所述第三指令集。

10.根据权利要求9所述的方法，其特征在于，所述根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集之前，还包括：

向所述CPU发送指令集回迁请求；

接收所述CPU发送的指令集回迁响应。

11.一种协处理器，应用于处理器系统，所述处理器系统包括所述协处理器和运行第一操作系统的中央处理器CPU，其特征在于，所述协处理器上运行第二操作系统；

所述协处理器包括：

第一指令集接收单元，用于接收所述CPU迁移的第一指令集，所述第一指令集用于指示所述CPU在所述第一操作系统中执行计算机操作，所述第一指令集为适用于所述第一操作系统的二进制代码的集合；

第二指令集获得单元，用于根据所述第一指令集获得第二指令集，其中，所述第二指令集中的二进制代码用于指示所述协处理器在所述第二操作系统中执行所述计算机操作；

所述第二指令集获得单元，还用于若所述第一指令集中的第三二进制代码的操作码在翻译表中未被匹配到，并且所述第三二进制代码为操作码属于第三部分子指令集的二进制代码，则将所述第三二进制代码做为所述第二指令集中的二进制代码，所述协处理器的指令集不包含所述第三部分子指令集，所述翻译表包含相同的计算机指令分别编译生成的在所述第一操作系统和所述第二操作系统中不同的操作码之间的对应关系；

第二指令集执行单元，用于执行所述第二指令集中的二进制代码。

12.根据权利要求11所述的协处理器，其特征在于，所述第二指令集获得单元，用于根据所述第一指令集获得第二指令集，包括：

所述第二指令集获得单元，用于在预先设置的所述翻译表中匹配所述第一指令集中的二进制代码的操作码，若所述第一指令集中的第一二进制代码的操作码在所述翻译表中被匹配到，则根据所述翻译表中所述第一二进制代码的操作码对应的匹配项，将所述第一二进制代码的操作码翻译为第二二进制代码的操作码，获得所述第二二进制代码，根据获得的至少一条所述第二二进制代码获得所述第二指令集，其中，所述翻译表包含相同的计算机指令分别编译生成的在所述第一操作系统和所述第二操作系统中不同的操作码之间的对应关系，所述第二二进制代码为适用于所述第二操作系统的二进制代码。

13.根据权利要求11所述的协处理器，其特征在于，所述协处理器还包括：

寄存器地址转换单元，用于将所述第二指令集包含的二进制代码中所述CPU的寄存器地址转换为所述协处理器的寄存器地址。

14.根据权利要求11所述的协处理器，其特征在于，所述第一指令集是所述CPU在所述CPU的CPU使用率大于第一阈值时向所述协处理器迁移的。

15.根据权利要求14所述的协处理器，其特征在于，所述第一指令集接收单元，用于接收所述CPU迁移的第一指令集，包括：

所述第一指令集接收单元，用于接收所述CPU发送的要迁移的所述第一指令集的地址，并基于所述第一指令集的地址访问所述CPU的内存来获取所述第一指令集；其中，所述第一指令集的地址是指所述第一指令集在所述CPU的内存中存储的地址，所述第一指令集的地址由所述CPU在所述CPU的内存使用率小于或等于第二阈值时向所述协处理器发送。

16.根据权利要求11所述的协处理器，其特征在于，所述第一指令集由所述CPU在所述CPU的内存使用率大于第二阈值时向所述协处理器发送。

17.根据权利要求11所述的协处理器，其特征在于，所述第二指令集执行单元，用于执行所述第二指令集中的二进制代码，包括：

所述第二指令集执行单元，用于依次执行所述第二指令集中的二进制代码；如果检测到执行所述第二指令集时出现二进制代码识别异常，则确定触发异常的第四二进制代码，将所述第四二进制代码转换为中间代码，再将所述中间代码转换为适用于所述第二操作系统的第五二进制代码，执行所述第五二进制代码，并继续执行所述第二指令集中所述第四二进制代码之后的二进制代码。

18.根据权利要求17所述的协处理器，其特征在于，所述第二指令集执行单元，还用于在将所述第四二进制代码转换为中间代码之前，向所述CPU发送指令集回迁请求，接收所述CPU发送的拒绝回迁指令。

19.根据权利要求11至16任一项所述的协处理器，其特征在于，所述第二指令集执行单元，用于执行所述第二指令集中的二进制代码，包括：

所述第二指令集执行单元，用于依次执行所述第二指令集中的二进制代码；如果检测到执行所述第二指令集时出现二进制代码识别异常，则确定触发异常的第六二进制代码；根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集，并向所述CPU迁移所述第三指令集。

20.根据权利要求19所述的协处理器，其特征在于，所述第二指令集执行单元，还用于

根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集之前，向所述CPU发送指令集回迁请求，接收所述CPU发送的指令集回迁响应。

21.一种协处理器，其特征在于，所述协处理器与存储器通过总线连接，所述存储器用于存储计算机执行指令，所述协处理器读取所述存储器存储的所述计算机执行指令，执行权利要求1至10任一项所述的计算机指令处理方法。

22.一种处理器系统，所述处理器系统包括中央处理器CPU和协处理器，其特征在于，所述CPU上运行第一操作系统，所述协处理器上运行第二操作系统；

所述CPU，用于向协处理器迁移第一指令集；

所述协处理器，用于执行权利要求1至10任一项所述的计算机指令处理方法。

# 计算机指令处理方法、协处理器和系统

## 技术领域

[0001] 本发明实施例涉及计算机领域，尤其涉及计算机指令处理方法、协处理器和系统。

## 背景技术

[0002] 目前，协处理器(coprocessor)，为一种芯片，主要用于代中央处理器(Central Processing Unit，简称CPU)处理特定任务。由于协处理器和中央处理器在指令集上的部分差异，在协处理器上运行的程序往往需要使用编译器单独编译，且需要对代码进行一定调整。现有技术中，对于某个应用的代码，一般会在该应用的代码中做好标签，通过该标签区分出哪些代码由CPU执行，哪些代码由协处理器执行。首先，对该应用的所有代码编译，包括根据标签对由CPU执行的代码和对由协处理器执行的代码进行不同的编译；代码编译后，CPU运行该应用的进程的过程中，如执行到需由协处理器执行的已编译代码，则CPU暂停执行该应用的进程并发送需由协处理器执行的已编译代码至协处理器，将标签处的代码卸载(offload)到协处理器上执行。

[0003] 从上可知，现有技术完成对应用的代码编译时，已明确哪部分已编译代码由CPU执行，哪些已编译代码由协处理器执行，不能根据需求将任意代码发送给协处理器执行，导致协处理器实际使用率较低，不能很好减轻CPU的负荷。

## 发明内容

[0004] 有鉴于此，本发明实施例提供了一种计算机指令处理方法、协处理器和系统，CPU可将计算机指令迁移至运行有操作系统的协处理器，由协处理器执行该计算机指令来减小CPU的负荷。

[0005] 第一方面，本发明实施例提供了一种计算机指令处理方法，应用于处理器系统，所述处理器系统包括协处理器和中央处理器CPU，所述CPU上运行第一操作系统，所述协处理器上运行第二操作系统；所述方法包括：

[0006] 所述协处理器接收所述CPU迁移的第一指令集，所述第一指令集用于指示所述CPU在所述第一操作系统中执行计算机操作，所述第一指令集为适用于所述第一操作系统的二进制代码的集合；

[0007] 所述协处理器根据所述第一指令集获得第二指令集，其中，所述第二指令集中的二进制代码用于指示所述协处理器在所述第二操作系统中执行所述计算机操作；

[0008] 所述协处理器执行所述第二指令集中的二进制代码。

[0009] 结合第一方面，在第一种可能的实现方式中，所述协处理器根据所述第一指令集获得第二指令集包括：

[0010] 所述协处理器在预先设置的翻译表中匹配所述第一指令集中的二进制代码的操作码，若所述第一指令集中的第一二进制代码的操作码在所述翻译表中被匹配到，则根据所述翻译表中所述第一二进制代码的操作码对应的匹配项，将所述第一二进制代码的操作码翻译为第二二进制代码的操作码，获得所述第二二进制代码，所述协处理器根据获得的

至少一条所述第二二进制代码获得所述第二指令集，其中，所述翻译表包含相同的计算机指令分别编译生成的在所述第一操作系统和所述第二操作系统中不同的操作码之间的对应关系，所述第二二进制代码为适用于所述第二操作系统的二进制代码。

[0011] 结合第一方面或者第一方面的的第一种可能的实现方式，在第二种可能的实现方式中，在所述协处理器执行所述第二指令集中的二进制代码之前，所述方法还包括：

[0012] 所述协处理器将所述第二指令集包含的二进制代码中所述CPU的寄存器地址转换为所述协处理器的寄存器地址。

[0013] 结合第一方面的第一种可能的实现方式或者第一方面的第二种可能的实现方式，在第三种可能的实现方式中，所述协处理器根据所述第一指令集获得第二指令集还包括：

[0014] 若所述第一指令集中的第三二进制代码的操作码在所述翻译表中未被匹配到，则所述协处理器将所述第三二进制代码做为所述第二指令集中的二进制代码。

[0015] 结合第一方面或者第一方面的的第一种可能的实现方式或者在第二种可能的实现方式或者第一方面的第三种可能的实现方式，在第四种可能的实现方式中，所述第一指令集是所述CPU在所述CPU的CPU使用率大于第一阈值时向所述协处理器迁移的。

[0016] 结合第一方面的第四种可能的实现方式，在第五种可能的实现方式中，所述协处理器接收所述CPU迁移的第一指令集，包括：

[0017] 所述协处理器接收所述CPU发送的要迁移的所述第一指令集的地址，所述第一指令集的地址是指所述第一指令集在所述CPU的内存中存储的地址，其中，所述第一指令集的地址由所述CPU在所述CPU的内存使用率小于或等于第二阈值时向所述协处理器发送；

[0018] 所述协处理器基于所述第一指令集的地址访问所述CPU的内存来获取所述第一指令集。

[0019] 结合第一方面或者第一方面的的第一种可能的实现方式或者在第二种可能的实现方式或者第一方面的第三种可能的实现方式，在第六种可能的实现方式中，所述第一指令集由所述CPU在所述CPU的内存使用率大于第二阈值时向所述协处理器发送。

[0020] 结合第一方面或者第一方面的的第一种可能的实现方式或者在第二种可能的实现方式或者第一方面的第三种可能的实现方式或者第一方面的的第四种可能的实现方式或者在第五种可能的实现方式或者第一方面的第六种可能的实现方式，在第七种可能的实现方式中，所述协处理器执行所述第二指令集中的二进制代码，包括：

[0021] 所述协处理器依次执行所述第二指令集中的二进制代码；

[0022] 如果检测到执行所述第二指令集时出现二进制代码识别异常，则确定触发异常的第四二进制代码；

[0023] 将所述第四二进制代码转换为中间代码，再将所述中间代码转换为适用于所述第二操作系统的第五二进制代码；

[0024] 执行所述第五二进制代码，并继续执行所述第二指令集中所述第四二进制代码之后的二进制代码。

[0025] 结合第一方面的第七种可能的实现方式，在第八种可能的实现方式中，所述将所述第四二进制代码转换为中间代码之前，还包括：

[0026] 向所述CPU发送指令集回迁请求；

[0027] 接收所述CPU发送的拒绝回迁指令。

[0028] 结合第一方面或者第一方面的的第一种可能的实现方式或者在第二种可能的实现方式或者第一方面的第三种可能的实现方式或者第一方面的的第四种可能的实现方式或者在第五种可能的实现方式或者第一方面的第六种可能的实现方式，在第九种可能的实现方式中，所述协处理器执行所述第二指令集中的二进制代码，包括：

[0029] 所述协处理器依次执行所述第二指令集中的二进制代码；

[0030] 如果检测到执行所述第二指令集时出现二进制代码识别异常，则确定触发异常的第六二进制代码；

[0031] 根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集，并向所述CPU迁移所述第三指令集。

[0032] 结合第一方面的第九种可能的实现方式，在第十种可能的实现方式中，所述根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集之前，还包括：

[0033] 向所述CPU发送指令集回迁请求；

[0034] 接收所述CPU发送的指令集回迁响应。

[0035] 第二方面，本发明实施例提供了一种协处理器，应用于处理器系统，所述处理器系统包括所述协处理器和运行第一操作系统的中央处理器CPU，所述协处理器上运行第二操作系统；所述协处理器包括：

[0036] 第一指令集接收单元，用于接收所述CPU迁移的第一指令集，所述第一指令集用于指示所述CPU在所述第一操作系统中执行计算机操作，所述第一指令集为适用于所述第一操作系统的二进制代码的集合；

[0037] 第二指令集获得单元，用于根据所述第一指令集获得第二指令集，其中，所述第二指令集中的二进制代码用于指示所述协处理器在所述第二操作系统中执行所述计算机操作；

[0038] 第二指令集执行单元，用于执行所述第二指令集中的二进制代码。

[0039] 结合第二方面，在第一种可能的实现方式中，所述第二指令集获得单元，用于根据所述第一指令集获得第二指令集，包括：

[0040] 所述第二指令集获得单元，用于在预先设置的翻译表中匹配所述第一指令集中的二进制代码的操作码，若所述第一指令集中的第一二进制代码的操作码在所述翻译表中被匹配到，则根据所述翻译表中所述第一二进制代码的操作码对应的匹配项，将所述第一二进制代码的操作码翻译为第二二进制代码的操作码，获得所述第二二进制代码，根据获得的至少一条所述第二二进制代码获得所述第二指令集，其中，所述翻译表包含相同的计算机指令分别编译生成的在所述第一操作系统和所述第二操作系统中不同的操作码之间的对应关系，所述第二二进制代码为适用于所述第二操作系统的二进制代码。

[0041] 结合第二方面或者第二方面的的第一种可能的实现方式，在第二种可能的实现方式中，所述协处理器还包括：

[0042] 寄存器地址转换单元，用于将所述第二指令集包含的二进制代码中所述CPU的寄存器地址转换为所述协处理器的寄存器地址。

[0043] 结合第二方面的第一种可能的实现方式或者第二方面的第二种可能的实现方式，在第三种可能的实现方式中，所述第二指令集获得单元，还用于若所述第一指令集中的第

三二进制代码的操作码在所述翻译表中未被匹配到，则所述协处理器将所述第三二进制代码做为所述第二指令集中的二进制代码。

[0044] 结合第二方面或者第二方面的的第一种可能的实现方式或者在第二种可能的实现方式或者第二方面的第三种可能的实现方式，在第四种可能的实现方式中，所述第一指令集是所述CPU在所述CPU的CPU使用率大于第一阈值时向所述协处理器迁移的。

[0045] 结合第二方面的第四种可能的实现方式，在第五种可能的实现方式中，所述第一指令集接收单元，用于接收所述CPU迁移的第一指令集，包括：

[0046] 所述第一指令集接收单元，用于接收所述CPU发送的要迁移的所述第一指令集的地址，并基于所述第一指令集的地址访问所述CPU的内存来获取所述第一指令集；其中，所述第一指令集的地址是指所述第一指令集在所述CPU的内存中存储的地址，其中，所述第一指令集的地址由所述CPU在所述CPU的内存使用率小于或等于第二阈值时向所述协处理器发送。

[0047] 结合第二方面或者第二方面的的第一种可能的实现方式或者在第二种可能的实现方式或者第二方面的第三种可能的实现方式，在第六种可能的实现方式中，所述第一指令集由所述CPU在所述CPU的内存使用率大于第二阈值时向所述协处理器发送。

[0048] 结合第二方面或者第二方面的的第一种可能的实现方式或者在第二种可能的实现方式或者第二方面的第三种可能的实现方式或者第二方面的的第四种可能的实现方式或者在第五种可能的实现方式或者第二方面的第六种可能的实现方式，在第七种可能的实现方式中，所述第二指令集执行单元，用于执行所述第二指令集中的二进制代码，包括：

[0049] 所述第二指令集执行单元，用于依次执行所述第二指令集中的二进制代码；如果检测到执行所述第二指令集时出现二进制代码识别异常，则确定触发异常的第四二进制代码，将所述第四二进制代码转换为中间代码，再将所述中间代码转换为适用于所述第二操作系统的第五二进制代码，执行所述第五二进制代码，并继续执行所述第二指令集中所述第四二进制代码之后的二进制代码。

[0050] 结合第二方面的第七种可能的实现方式，在第八种可能的实现方式中，所述第二指令集执行单元，还用于在将所述第四二进制代码转换为中间代码之前，向所述CPU发送指令集回迁请求，接收所述CPU发送的拒绝回迁指令。

[0051] 结合第二方面或者第二方面的的第一种可能的实现方式或者在第二种可能的实现方式或者第二方面的第三种可能的实现方式或者第二方面的的第四种可能的实现方式或者在第五种可能的实现方式或者第二方面的第六种可能的实现方式，在第九种可能的实现方式中，所述第二指令集执行单元，用于执行所述第二指令集中的二进制代码，包括：

[0052] 所述第二指令集执行单元，用于依次执行所述第二指令集中的二进制代码；如果检测到执行所述第二指令集时出现二进制代码识别异常，则确定触发异常的第六二进制代码；根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集，并向所述CPU迁移所述第三指令集。

[0053] 结合第二方面的第九种可能的实现方式，在第十种可能的实现方式中，所述第二指令集执行单元，还用于根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集之前，向所述CPU发送指令集回迁请求，接收所述CPU发送的指令集回迁响应。

[0054] 第三方面，本发明实施例提供了一种协处理器，所述协处理器与存储器通过总线连接，所述存储器用于存储计算机执行指令，所述协处理器读取所述存储器存储的所述计算机执行指令，执行上述第一方面或者第一方面的任一种可能的实施方式提供的计算机指令处理方法。

[0055] 第四方面，本发明实施例提供了一种处理器系统，所述处理器系统包括中央处理器CPU和协处理器，其特征在于，所述CPU上运行第一操作系统，所述协处理器上运行第二操作系统；

[0056] 所述CPU，用于向协处理器迁移第一指令集；

[0057] 所述协处理器，用于上述第一方面或者第一方面的任一种可能的实施方式提供的计算机指令处理方法。

[0058] 通过上述方案，对于编译所得的适于在第一操作系统执行的第一指令集，协处理器根据第一指令集获取在协处理器执行的第二指令集，由协处理器执行第二指令集代替由CPU执行第一指令集，减小CPU的负荷，提高协处理器的使用率。

## 附图说明

[0059] 图1为现有技术分配编译后的二进制代码的应用场景的一种系统逻辑结构示意图；

[0060] 图2为计算机指令处理方法的应用场景的一种系统逻辑结构示意图；

[0061] 图3为中央处理器的指令集中的计算机指令与协处理器的指令集的对应关系的示意图；

[0062] 图4为计算机指令处理方法的一种示范性流程图；

[0063] 图5为步骤A402的一种可选示范性流程图；

[0064] 图6为计算机指令处理方法的又一种示范性流程图；

[0065] 图7为步骤A401的一种可选示范性流程图；

[0066] 图8为步骤A403中处理二进制代码异常的一种可选示范性流程图；

[0067] 图9为步骤A403中处理二进制代码异常的一种可选示范性流程图；

[0068] 图10为步骤A403中处理二进制代码异常的一种可选示范性流程图；

[0069] 图11为步骤A403中处理二进制代码异常的一种可选示范性流程图；

[0070] 图12为协处理器202的一种逻辑结构示意图；

[0071] 图13为协处理器202的一种可选逻辑结构示意图；

[0072] 图14为协处理器1401与存储器1402组成的系统的系统逻辑结构示意图。

## 具体实施方式

[0073] 下面将结合本发明实施例中的附图，对本发明实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例是本发明一部分实施例，而不是全部的实施例。基于本发明中的实施例，本领域普通技术人员在没有作出创造性劳动前提下所获得的所有其他实施例，都属于本发明保护的范围。

[0074] 参见图1，现有技术提供的系统100，该系统100包括中央处理器(Central Processing Unit，简称CPU)101和协处理器102，中央处理器101支持的指令集与协处理器

102支持的指令集不同。另外，中央处理器101安装有操作系统(例如，支持X86指令集的操作系统)，协处理器102没有安装操作系统。其中，中央处理器101运行的程序在对源代码编译时，针对由协处理器102处理的源代码根据协处理器102的指令集编译，编译所得的二进制代码对于协处理器102来说是可识别的和可执行的，但对于中央处理器101来说可能存在不可识别的二进制代码；针对由中央处理器101处理的源代码根据中央处理器101的指令集编译，编译所得的二进制代码对于中央处理器101来说是可识别的和可执行的，但对于协处理器102来说可能存在不可识别的二进制代码，因此即使中央处理器101的负荷大，协处理器102也不能分担执行由中央处理器101执行的二进制代码。

[0075] 参见图2，本发明实施例提供的系统200，该系统200包括中央处理器201和协处理器202。协处理器202具有控制能力，中央处理器201和协处理器202各自运行操作系统；相对于现有技术中协处理器202没有安装操作系统，本发明在协处理器202安装操作系统，可在协处理器202运行进程，并由协处理器202的操作系统进行进程调度；这样，中央处理器201和协处理器202之间可以相互迁移进程。

[0076] 可选地，该系统200可以同时位于同一数据处理设备，对于该系统200包括的中央处理器201和协处理器202在该数据处理设备上设置的位置，本发明实施例不做限定；该中央处理器201与该协处理器202连接，对于该系统200包括的中央处理器201和协处理器202在该数据处理设备上如何实现连接的，本发明实施例不做限定；连接后的该中央处理器201与该协处理器202可相互之间进行数据传输。

[0077] 例如，该数据处理设备包括总线，系统200包括的中央处理器201和协处理器202同时连接在该总线上，中央处理器201与协处理器202之间经该总线进行数据交互，在该总线满足中央处理器201与协处理器202之间的数据传输需求的情况下，该数据传输需求包括数据传输速度和数据传输格式，对于该总线的具体型号和支持的总线协议不做限定；另外，随着时代发展，可采用其他介质连接该系统200包括的中央处理器201和协处理器202，通过该介质提高该中央处理器201与该协处理器202之间的数据交互速度，具体实施时，可将该介质替代用于连接该中央处理器201与该协处理器202的总线，或者该介质与该总线并存，都用于该中央处理器201与该协处理器202之间的数据交互。

[0078] 举例说明，该系统200中，中央处理器201采用X86处理器实现，协处理器202采用因特尔集成众核架构(Many Integrated Core，简称MIC)实现，该X86处理器和该MIC通过快速外设部件互连(Peripheral Component Interconnect Express，简称PCI-E)总线连接，X86处理器和MIC之间通过PCI-E总线进行数据交互。

[0079] 可选地，系统200中，中央处理器201与协处理器202不位于同一设备内，该中央处理器201与该协处理器202通信连接，该中央处理器201与该协处理器202之间通过消息的方式进行数据交互。

[0080] 参见图2，在本发明提供的计算机指令处理方法的应用场景中，中央处理器201支持的指令集与协处理器202支持的指令集会包含部分不同的计算机指令，中央处理器201运行的第一操作系统支持中央处理器201的指令集，协处理器202运行的操作系统支持协处理器202的指令集；如果第一操作系统与第二操作系统相同，则第一操作系统与第二操作系统均同时支持中央处理器201的指令集和协处理器202的指令集；如果第一操作系统不支持协处理器202的指令集或者第二操作系统不支持中央处理器201的指令集，则第一操作系统与

第二操作系统是不相同的操作系统。

[0081] 相对于协处理器202支持的指令集，中央处理器201支持的指令集可划分为三部分子指令集，包括第一部分子指令集、第二部分子指令集和第三部分子指令集。

[0082] 第一部分子指令集，其包含的每个计算机指令，协处理器202的指令集中也包含相同的计算机指令；并且，中央处理器201支持的表示该计算机指令的二进制码，与协处理器202支持的表示该计算机指令的二进制码相同。以图3为例，二进制码“AAAA”表示第一部分子指令集中的一个计算机指令，协处理器支持的指令集也包含该计算机指令，并且，该二进制码“AAAA”也表示协处理器支持的指令集中的该计算机指令；与二进制码“AAAA”类似，对应表示第一部分子指令集中其他两个计算机指令的二进制码“CCCC”和二进制码“DDDD”，也表示协处理器的指令集中包含的该两个计算机指令。

[0083] 第二部分子指令集，其包含的每个计算机指令，协处理器202的指令集中也包含相同的计算机指令；但是，中央处理器201支持的表示该计算机指令的二进制码，与协处理器202支持的表示该计算机指令的二进制码是不相同的。以图3为例，二进制码“BBBB”表示第二部分子指令集中的某个计算机指令，协处理器支持的指令集也包含该计算机指令，但是，协处理器支持的表示该计算机指令的二进制码“B1B1B1B1”，“BBBB”与“B1B1B1B1”为不同的二进制码。

[0084] 第三部分子指令集，其包含的每个计算机指令，协处理器202的指令集并不包含。以图3为例，二进制码“EEEE”表示第三部分子指令集中的某个计算机指令，协处理器的指令集并不包含该计算机指令，因此对于二进制码“EEEE”，协处理器的指令集中找不到对应该计算机指令的二进制码。

[0085] 本发明实施例中，若中央处理器201向协处理器202迁移进程，协处理器202接收中央处理器201迁移的与该进程相关的数据，包括执行该进程所需的二进制代码，还包括该进程的进程状态等等。在执行该进程所需的每条二进制代码中，该二进制代码的操作码属于表示计算机指令的二进制码，该二进制代码还可能包括操作数，操作数也是采用二进制码表示的。

[0086] 对于第二部分子指令集中的计算机指令，中央处理器201支持的表示该计算机指令的二进制码与协处理器202支持的表示该计算机指令的二进制码是不相同的；鉴于此，本发明提供的计算机指令处理方法针对第二部分子指令集建立翻译表，通过翻译表进行二进制码的转换，将中央处理器201支持的表示该第二部分子指令集中计算机指令的二进制码转换为协处理器202支持的表示该计算机指令的二进制码，这样协处理器202能够识别转换后的二进制码，运行迁移后的进程的过程中执行该转换后的二进制码实现计算机指令所具有的功能。对于第三部分子指令集包含的每个计算机指令，本发明提供的计算机指令处理方法都会出现无法识别表示该计算机指令的二进制码，出现无法识别二进制代码(该二进制代码的操作码为表示该计算机指令的二进制码)的异常，这时有两种解决途径：

[0087] 第一种，将进程回迁中央处理器201；

[0088] 第二种，将触发异常的二进制代码先转换为一个或多个中间代码，再将每个中间代码转换为协处理器202支持的二进制代码，从转换后的二进制代码开始继续运行进程；一种经中间代码转换的可选具体实现是：根据触发异常的二进制代码中的操作码(属于表示第三部分子指令集中计算机指令的二进制码)确定每个中间代码表示的操作码，如果还有

中间代码表示的操作数，再根据触发异常的二进制代码中的操作数确定中间代码表示的与每个操作码对应的操作数，然后再根据中间代码表示的操作码确定协处理器202支持的二进制代码中的每个操作码，并根据中间代码表示的操作数确定协处理器202支持的二进制代码中的每个操作码对应的操作数。

[0089] 可选地，将所述中央处理器201设置为首选执行进程的设备，设置协处理器202为次选执行进程的设备，首先由中央处理器201执行进程，执行进程的过程中如出现以下情况，则将进程迁移至协处理器202，包括：

[0090] 第一种情况，中央处理器201执行进程的过程中，识别到由协处理器202负责执行的二进制代码时，将该进程迁移至协处理器202，由协处理器202运行该进程来执行该二进制代码；可选地，协处理器202将该进程的执行结果反馈中央处理器201；

[0091] 第二种情况，当中央处理器201的CPU使用率过高时，筛选出一个或多个进程(例如筛选出占CPU使用率最大的进程)，将筛选出的进程迁移至协处理器202，由协处理器202根据翻译表对执行该进程所需的二进制代码进行翻译、再执行翻译后的二进制代码来运行该进程；其中，翻译动作具体为，对执行进程所需的二进制代码遍历地进行匹配查找，可选地是按照二进制代码的执行顺序依次遍历地匹配查找，匹配查找是否存在翻译表记录的中央处理器201支持的二进制码，若匹配查找到，根据翻译表将查找到的二进制码替换为表示相同计算机指令的、协处理器202支持的二进制码；

[0092] 第三种情况，当中央处理器201使用内存的内存使用率过高时，筛选出一个或多个进程(例如筛选出内存使用率最大的进程)，将筛选出的进程迁移至协处理器202，由协处理器202根据翻译表对执行该进程所需的二进制代码进行翻译、再执行翻译后的进程。

[0093] 本发明一实施例，详述协处理器如何运行从CPU迁移过来的进程，为便于说明，以协处理器采用MIC实现、以中央处理器采用X86处理器实现为例，X86处理器与MIC通过PCI-E总线连接，X86处理器运行通用操作系统(如支持X86指令集的操作系统)，MIC运行定制的uOS操作系统，X86处理器和MIC端分别具有独立的内存和寄存器。

[0094] 本实施例中，X86处理器的寄存器为128位的寄存器，例如X86处理器包含16个128位的XMM寄存器，该XMM寄存器属于矢量寄存器，支持单指令流多数据流扩展(Streaming SIMD Extensions，简称SSE)指令集，其中，所述SIMD的中文名称为单指令流多数据流，所述SIMD的英文全称为Single Instruction Multiple Data；即X86处理器可采用SSE指令集操作XMM寄存器组，进行128位的矢量运算。或者，X86处理器的寄存器为256位的寄存器，例如X86处理器包含16个256位的YMM寄存器，该YMM寄存器属于矢量寄存器，支持高级矢量扩展(Advanced Vector Extensions，简称AVX)指令集，即X86处理器可采用AVX指令集操作YMM寄存器组，进行256位的矢量运算，例如操作YMM寄存器组进行浮点数的运算。

[0095] MIC的寄存器为512位的寄存器，例如MIC包含32个512位的ZMM寄存器，该ZMM寄存器属于矢量寄存器，可操作ZMM寄存器组进行512位的矢量运算；另外，MIC的寄存器也支持SSE指令集和AVX指令集。

[0096] 本实施例中，MIC为兼容X86处理器操作寄存器的运算，MIC选用32个寄存器中的16个寄存器，支持128位的运算(如支持SSE指令集的运算)，和支持256位的运算(如支持AVX指令集的的运算)；可选地，MIC使用16个512位的寄存器中低256位，进行128位的运算或者进行256位的运算，来兼容X86处理器操作寄存器的运算。这样，针对X86处理器的迁移的进程，

MIC将执行该进程所需的二进制代码中代表X86处理器的寄存器的二进制码替换为代表MIC中选用的寄存器的二进制码，即将指向X86处理器的寄存器的二进制码替换为指向MIC中选用的寄存器的二进制码，不需对运算数据进行位数调整和进行运算规则的调整，即可使用MIC的寄存器进行运算；例如，MIC将X86处理器迁移的二进制代码中代表X86处理器的XMM寄存器的二进制码替换为代表MIC中选用的寄存器的二进制码，MIC执行表示SSE指令集中的计算机指令的二进制代码，操作MIC中选用的16个寄存器进行128位的矢量运算；再例如，MIC将X86处理器迁移的二进制代码中代表X86处理器的YMM寄存器的二进制码替换为代表MIC中选用的寄存器的二进制码，MIC执行表示AVX指令集中的计算机指令的二进制代码，操作MIC中选用的16个寄存器进行256位的矢量运算。

[0097] 这样，对于X86处理器向MIC迁移的进程，MIC将执行该进程所需的二进制代码中代表X86处理器的寄存器的二进制码替换为代表MIC中选用的寄存器的二进制码后，使用MIC的寄存器运行迁移的进程。

[0098] 另外在本实施例中，建立翻译表，建立的翻译表由MIC执行匹配。具体地，翻译表是针对上述第二部分子指令集包含的计算机指令建立的，原因是：X86处理器支持的表示该计算机指令的二进制码与MIC支持的表示该计算机指令的二进制码不同；建立翻译表的方法为，在翻译表中分别添加X86处理器支持的表示该计算机指令的二进制码、MIC支持的表示该计算机指令的二进制码，并在翻译表中确定X86处理器支持的表示该计算机指令的二进制码与MIC支持的表示该计算机指令的二进制码的映射关系。作为翻译表一个举例，表1例举了五个计算机指令，如下：

[0099] 表1

[0100]

| 计算机指令 | X86 处理器支持的二进制码 | MIC 支持的二进制码 |
|---|---|---|
| FXSAVE | “01111011110111010000” | “00001111101011101111” |
| FXRSTOR | “11011101” | “00001111101011100001” |
| RDPMC | “0000111100110001”和“0000111100110010” | “0000111100110011” |
| FSUB | “1101100011100000” | “0000111101011100” |

[0101] 表1中的“FXSAVE”，是指示浮点保存状态的指令，用于保存浮点运算单元(Float Point Unit，简称FPU)寄存器的状态，X86处理器支持的表示“FXSAVE”的二进制码为“01111011110111010000”，MIC支持的表示“FXSAVE”的二进制码为“00001111101011101111”；

[0102] 表1中的“FXRSTOR”，是指示浮点恢复状态的指令，用于将保存的FPU寄存器的状态恢复到FPU寄存器中；X86支持的表示“FXRSTOR”的二进制码为“11011101”，MIC支持的表示“FXRSTOR”的二进制码为“00001111101011100001”；

[0103] 表1中的“RDPMC”，是读执行监视计数的指令，用于读取性能监控器的计数；X86处理器支持的表示“RDPMC”的二进制码有两个，包括“0000111100110001”和“0000111100110010”，MIC支持的表示“RDPMC”的二进制码仅一个，为“0000111100110011”；

[0104] 表1中的“FSUB”，是浮点减指令，用于浮点数的减法运算，X86支持的表示“FSUB”的二进制码为“1101100011100000”，MIC支持的表示“FSUB”的二进制码为“0000111101011100”。

[0105] 这样，MIC运行的uOS操作系统加载该翻译表后，针对X86处理器向MIC迁移进程的过程中MIC从CPU获取到的与该进程相关的数据，从与该进程相关的数据提取出执行该进程所需的二进制代码，根据该翻译表对该二进制代码中与第二部分子指令集包含的计算机指令对应的二进制码进行匹配替换，替换为翻译表中MIC支持的表示该计算机指令的二进制码，继而MIC能够识别替换后的二进制码，一定程度提高了MIC执行迁移后的进程的正确率和效率。

[0106] 在本实施例中，触发X86处理器向MIC迁移进程的条件有两个，一个条件是检测到X86处理器的CPU使用率大于第一阈值；又一个条件是检测到X86处理器使用内存的内存使用率大于第二阈值。具体实现时，X86处理器的操作系统运行一段代码来实现一监控模块，通过该监控模块检测X86处理器的负荷，包括：检测X86处理器的CPU使用率，和检测X86处理器使用内存的内存使用率。该监控模块检测到以上两个条件中的任一得到满足时，将某一个或多个进程挂起，锁定该进程使用的内存空间，向MIC迁移该进程。

[0107] X86处理器将进程向MIC迁移时，如果是因X86处理器使用内存的内存使用率大于第二阈值触发的将进程从X86处理器迁移到MIC，无论X86处理器的CPU使用率是否大于第一阈值，都将X86处理器使用的内存中与该进程相关的数据发送给MIC。对应地，MIC将接收的与该进程相关的数据存储到MIC使用的内存中，再从MIC的内存存储的与该进程相关的数据中提取执行该进程所需的二进制代码；若在遍历提取的二进制代码中匹配到翻译表中CPU支持的二进制码(与第二部分子指令集包含的计算机指令对应的二进制码)，替换为翻译表中MIC支持的标识该计算机指令的对应二进制码，并以替换的二进制码更新存储至MIC的内存；这样，对于MIC的内存中存储的与第二部分子指令集包含的计算机指令对应的二进制码，都会根据翻译表替换为MIC支持的表示该计算机指令的二进制码。

[0108] X86处理器将进程向MIC迁移时，如果是因X86处理器的CPU使用率大于第一阈值且X86处理器使用内存的内存使用率小于或等于第二阈值而触发的将进程从X86处理器迁移到MIC，将X86处理器的内存中存储该进程相关的数据的存储地址发送至MIC。对应地，MIC从其使用的内存中划分出一个存储空间，建立该个存储空间包含的存储地址与接收的存储地址(X86处理器的内存中存储该进程相关的数据的存储地址)的地址映射关系；继而，MIC通过PCI-E总线，根据该地址映射关系访问X86处理器的内存中与该进程相关的数据，从与该进程相关的数据中提取执行该进程所需的二进制代码；若在提取的二进制代码中匹配到翻译表中CPU支持的二进制码(即与第二部分子指令集包含的计算机指令对应的二进制码)，替换为翻译表中MIC对应支持的二进制码，并将替换的二进制码更新存储至X86处理器的内存。这样，X86处理器的内存中存储的与第二部分子指令集包含的计算机指令对应的二进制码，都会根据翻译表替换为MIC支持的表示该计算机指令的二进制码；进而，MIC根据该地址映射关系使用CPU的内存来运行该进程。

[0109] 下面，对MIC如何根据翻译表转换二进制码的具体实现做如下描述：

[0110] MIC的uOS操作系统运行一段代码来实现翻译模块，该翻译模块加载翻译表。

[0111] 针对X86处理器迁移来的每个进程，翻译模块遍历执行该进程所需的二进制代码，

匹配查找是否存在翻译表中X86处理器支持的二进制码，每次匹配查找到，根据翻译表将该二进制码翻译成MIC支持的二进制码，直到完成遍历查找。

[0112] 例如，对于异或指令(XOR)，X86处理器支持的二进制码根据比较的对象不同而有所变化，如果比较两个寄存器值，X86处理器支持该异或指令的二进制码表示为“0011001”，如果比较寄存器值和内存存储的值，X86处理器支持该异或指令的二进制码表示为“0011000”；但是，MIC支持该异或指令(XOR)的二进制码统一表示为“0011000”；为MIC能够识别比较两个寄存器值的该异或指令，在翻译表中记录“0011001”与“0011000”的映射关系，若翻译模块从X86处理器迁移的进程包含的二进制代码中根据翻译表匹配查找到“0011001”，便根据该翻译表将该进程包含的二进制码中的“0011001”替换为“0011000”。

[0113] 再例如，对于用于读取性能监控器的计数的“RDPMC”指令，X86处理器支持该“RDPMC”指令的二进制码为“0000111100110001”和“0000111100110010”，MIC支持的表示“RDPMC”的二进制码仅一个，为“0000111100110011”，因此在翻译表中记录“0000111100110001”与“0000111100110011”的映射关系，和在翻译表中记录“0000111100110010”与“0000111100110011”的映射关系。若翻译模块从X86处理器迁移的进程所包含的二进制代码中根据翻译表匹配查找到“0000111100110001”，便根据该翻译表将该进程包含的二进制码中的“0000111100110001”替换为“0000111100110011”；若翻译模块从X86处理器迁移的进程包含的二进制代码中根据翻译表匹配查找到“0000111100110010”，便根据该翻译表将该进程包含的二进制码中的“0000111100110010”替换为“0000111100110011”。

[0114] 本实施例中，X86处理器向MIC迁移进程的过程中，MIC不但会从X86处理器使用的内存获取与该进程相关的数据，还会从X86处理器的寄存器获取与该进程相关的寄存器值，并将获取的寄存器值转存储至MIC的对应寄存器。MIC运行进程时，首先从MIC的内存存储的与该进程相关的数据中提取该进程的进程状态，该进程状态包括进程优先级、进程标识符、栈指针等进程运行的必要状态信息；然后，MIC从根据该进程状态确定的进程运行节点开始，使用MIC的寄存器，基于MIC的内存中存储的与该进程相关的数据(该数据包括根据翻译表翻译后的二进制代码)，执行进程。

[0115] 本实施例中，MIC的uOS操作系统运行一段代码来实现异常处理模块，该异常处理模块能够截获MIC执行进程出现的异常，包括进程执行到不能识别的二进制代码所触发的异常。可选地，MIC执行进程的过程中，如果异常处理模块检测到进程执行异常，将进程挂起，并生成记录进程异常执行的异常信息。

[0116] 例如，上述第三部分子指令集包含的计算机指令的二进制码，MIC的指令集没有包含有，MIC无法识别与该计算机指令对应的二进制码；另外，翻译表也没有记录该第三部分子指令集包含的计算机指令的二进制码，无法根据翻译表将该第三部分子指令集包含的计算机指令的二进制码转换为MIC的指令集中的计算机指令对应的二进制码；因此，MIC不能识别以该二进制码作为操作码的二进制代码，如果进程执行到该二进制代码便会触发二进制代码无法识别的异常。

[0117] 异常处理模块检测到因指令识别异常触发的执行进程异常，将进程挂起，采用以下三种可选的异常处理方式进行异常处理：

[0118] 第一种方式，异常处理模块从触发进程异常的二进制代码开始将该挂起的进程回

迁X86处理器，具体包括：对内存(可能是MIC的内存，或者是X86处理器的内存)中执行该回迁进程所需的二进制代码根据翻译表进行操作码(采用二进制码表示)的匹配，将匹配到的二进制码转换为X86处理器支持的二进制码，以转换的二进制码更新该内存中对应的二进制码。如果是在MIC的内存存储的更新后的执行该回迁进程所需的二进制代码，将该更新后的执行该回迁进程所需的二进制代码转存储至X86处理器的内存，一种转存储的实现方式是，通过MIC与X86处理器的数据通信，将MIC的内存中存储的更新后的执行该回迁进程所需的二进制代码转存储至X86处理器的内存。另外还将执行该回迁进程所需的二进制代码中表示MIC的寄存器的二进制码替换为表示X86处理器的寄存器的二进制码，并将替换后的二进制码更新该内存中存储的表示MIC的寄存器的二进制码；另外还将MIC的寄存器中存储的与该回迁进程相关的寄存器值转存储至X86处理器的寄存器中；这样，X86处理器可以使用其寄存器和其内存运行该回迁进程。

[0119] 第二种方式，异常处理模块判断该挂起的进程是否属于X86处理器迁移到MIC的进程，如果该挂起的进程属于X86处理器迁移到MIC的进程，识别触发异常的二进制代码，并将触发异常的二进制代码转换为模拟器(如simics模拟器或者qemu模拟器)的中间代码，再将该中间代码转换为MIC支持的二进制代码，从转换得到的二进制代码开始继续执行该进程；

[0120] 第三种方式，异常处理模块向X86处理器发送进程回迁请求来通知X86处理器期望回迁当前异常执行的进程；X86处理器响应该进程回迁请求，并确定：监控模块在当前监控到的X86处理器的使用率、在当前监控到的X86处理器的内存的内存使用率；如果X86处理器确定的结果是X86处理器的使用率小于第一阈值、且X86处理器的内存的内存使用率小于或等于第二阈值，X86处理器向MIC反馈进程回迁指令；如果X86处理器确定的结果是X86处理器的使用率大于第一阈值，或者如果X86处理器确定的结果是X86处理器的内存的内存使用率大于第二阈值，X86处理器向MIC反馈进程拒绝回迁指令；

[0121] 在第三种方式中，如果异常处理模块接收到X86处理器反馈的进程回迁指令，将挂起的进程回迁X86处理器，将进程从MIC回迁X86处理器的实现方式与上述第一种方式同原理实现，在此不再赘述；

[0122] 在第三种方式中，如果异常处理模块接收到X86处理器反馈的拒绝回迁指令，不将挂起的进程回迁X86处理器，并判断该挂起的进程是否属于X86处理器迁移到MIC的进程，如果该挂起的进程属于X86处理器迁移到MIC的进程，识别触发异常的二进制代码，并将触发异常的二进制代码转换为模拟器(如simics模拟器或者qemu模拟器)的中间代码，再将该中间代码转换为MIC支持的二进制代码，从转换得到的二进制代码开始继续执行该进程。可选地，如果该挂起的进程不属于X86处理器迁移到MIC的进程，MIC直接输出异常信息；其中，该挂起的进程不属于X86处理器迁移到MIC的进程的原因可能是：X86处理器具有一进程，执行该进程的过程中识别到需由MIC执行的代码段，将该代码段转由MIC执行，MIC新建立一进程来执行该代码段并出现异常。

[0123] 本实施例中，X86处理器支持的以下计算机指令，MIC是不支持的，包括以下22种计算机指令：条件跳转指令“CMOV”、比较交换16字节指令“CMPXCHG16B”、浮点条件跳转指令“FCMOVcc”、浮点比较加载标志指令“FCOMI”、浮点比较加载标志出栈指令“FCOMIP”、浮点反比加载标志指令“FUCOMI”、浮点反比加载标志出栈指令“FUCOMIP”、端口输入指令“IN”、端口输入串指令“INS”、端口输入字节串指令“INSB”、端口输入双字串指令“INSD”、端口输入

字串指令“INSW”、监视指令“MONITOR”、线程同步指令“MWAIT”、端口输出指令“OUT”、端口输出串指令“OUTS”、端口输出字节串指令“OUTSB”、端口输出双字指令“OUTSD”、端口输出字串指令“OUTSW”、暂停指令“PAUSE”、系统进入指令“SYSENTER”、系统退出指令“SYSEXIT”;这22种计算机指令可以分为三大类:

[0124] 第一类是可拆成两个动作的计算机指令,包括条件跳转指令“CMOV”、比较交换16字节指令“CMPXCHG16B”、浮点条件跳转指令“FCMOVcc”、浮点比较加载标志指令“FCOMI”、浮点比较加载标志出栈指令“FCOMIP”、浮点反比加载标志指令“FUCOMI”、浮点反比加载标志出栈指令“FUCOMIP”;若MIC执行包含第一类计算机指令的二进制代码导致进程发生异常,如需MIC继续执行该进程,MIC将包含第一类计算机指令的二进制代码转换为中间代码,该中间代码为包含两个动作的二进制代码,这两个动作在MIC的指令集中分别存在对应的计算机指令,再将该中间代码转换为MIC支持的二进制代码,该MIC支持的二进制代码中每个二进制代码的操作码分别表示MIC的指令集中对应的计算机指令,这样,MIC能够识别转换后的二进制码并执行。以条件跳转指令“CMOV”为例,MIC不能识别以表示该条件跳转指令的二进制码为操作数的二进制代码,异常处理模块将该二进制代码经中间代码转换为MIC支持的二进制代码,包括指令在二进制码形式下的以下转换:根据该条件跳转指令确定中间代码表示的条件判断指令和移动指令,再将该中间代码表示的条件判断指令和移动指令对应翻译成MIC可识别的条件判断指令和移动指令(MOV);MIC先执行条件判断指令确定是否满足跳转条件,如果满足,再执行移动指令(MOV);

[0125] 第二类是通过端口读取数据或者写入数据的计算机指令,包括端口输入指令“IN”、端口输入串指令“INS”、端口输入字节串指令“INSB”、端口输入双字串指令“INSD”、端口输入字串指令“INSW”、监视指令“MONITOR”、线程同步指令“MWAIT”、端口输出指令“OUT”、端口输出串指令“OUTS”、端口输出字节串指令“OUTSB”、端口输出双字指令“OUTSD”、端口输出字串指令“OUTSW”;MIC执行第二类指令导致进程发生异常,如果继续由MIC执行该进程,分两种情况处理;第一种情况是通过端口写入数据,该情况下,MIC通知X86处理器从该计算机指令指定的目标端口写入数据即可;第二种情况是读取数据的计算机指令,该情况下,MIC先通知X86处理器从该计算机指令指定的目标端口读取数据到内存中,再访问这块内存来获取该数据;

[0126] 第三类包括暂停指令“PAUSE”、系统进入指令“SYSENTER”、系统退出指令“SYSEXIT”,这三条指令是后期添加的,为了增加性能,其中暂停指令“PAUSE”是为了减少自旋锁的性能损失,系统进入指令“SYSENTER”、系统退出指令“SYSEXIT”是为了减少内核态和用户态之间切换的损失;第三类计算机指令是对原有X86指令集的优化,但MIC并不支持这种优化;若MIC执行二进制码表示的暂停指令“PAUSE”触发进程异常,如果继续由MIC执行该进程,执行二进制码表示的自旋锁指令,来替代执行二进制码表示的暂停指令“PAUSE”,即可继续运行进程;若执行二进制码表示的系统进入指令“SYSENTER”触发进程异常,执行二进制码表示的切换用户态到内核态的切换指令,来替代执行二进制码表示的系统进入指令“SYSENTER”,即可继续运行进程;若执行二进制码表示的系统退出指令“SYSEXIT”触发进程异常,执行二进制码表示的切换内核态到用户态的切换指令,来替代执行二进制码表示的系统退出指令“SYSEXIT”,即可继续运行进程。

[0127] 本实施例中,如果MIC顺利执行完X86处理器迁移的进程,MIC可将执行结果反馈

X86处理器，也可控制直接输出执行结果，输出的方式包括但不限于：通过显示模块等数据输出模块呈现执行结果，或者基于执行结果执行其他动作。

[0128] 本实施例中，MIC运行uos操作系统后可调度进程，继而，X86处理器在负荷较大(X86处理器的使用率大于第一阈值，和/或X86处理器的内存的使用率大于第二阈值)时将筛选的进程迁移至MIC执行，分担了X86处理器的负荷；尤其是，对于大负荷的进程，X86处理器可将该进程迁移至MIC执行，延长了X86处理器的使用寿命，还尽量保证了各进程都能分配到足够的资源，保证了各进程的执行效率。

[0129] 本发明一实施例，基于上述的系统200及上述实施例对协处理器如何运行从CPU迁移过来的进程的所作的改进，本实施例对上述实施例的技术方案做适应扩展，从协处理器101的角度提出一种计算机指令处理方法实现的基础流程，图4为该计算机指令处理方法的一种示范性的工作流程，但为了便于描述，仅示出了与本发明实施例相关的部分。

[0130] 本实施例提供的计算机指令处理方法，应用于处理器系统，所述处理器系统包括协处理器和中央处理器CPU，所述CPU上运行第一操作系统，所述协处理器上运行第二操作系统；其中，第一操作系统是指支持CPU的指令集的操作系统；第二操作系统是指支持协处理器的指令集的操作系统。

[0131] 所述协处理器运行第二操作系统后，可通过第二操作系统运行进程和线程，进行进程之间的调度，及进行线程之间的调度；进而，CPU可以向协处理器迁移一个或多个进程，本实施例定义第一进程为从CPU向协处理器迁移的单个进程；另外，CPU还可以向协处理器迁移一个或多个线程，本实施例定义第一线程为从CPU向协处理器迁移的单个线程。

[0132] 更进一步地，CPU与协处理器之间不但可以相互迁移进程，相互迁移线程，还可以相互迁移一条或多条二进制代码。本实施例定义第一指令集，如果CPU向协处理器迁移进程，第一指令集是指执行该进程所需的二进制代码；如果CPU向协处理器迁移线程，第一指令集是指执行该线程所需的二进制代码；如果CPU向协处理器迁移一条或多条二进制代码，第一指令集是指CPU迁移协处理器的二进制代码的集合。

[0133] 第一指令集包含的二进制代码，是根据CPU的指令集对源代码编译得到，对是否由第一操作系统执行的该编译不做限定，可以是第一操作系统执行的该编译，也可以是其他编译器完成该编译后第一操作系统再获得该第一指令集，另外此处对源代码不做限定，并对采用哪种编程语言编写得到该源代码不做限定。

[0134] 如图4所示，本实施例提供的计算机指令处理方法包括：步骤A401、步骤A402和步骤A403。

[0135] 步骤A401，所述协处理器接收所述CPU迁移的第一指令集，所述第一指令集用于指示所述CPU在所述第一操作系统中执行计算机操作，所述第一指令集为适用于所述第一操作系统的二进制代码的集合。

[0136] 本实施例中，对触发CPU向协处理器迁移第一指令集的触发条件不做限定，甚至CPU可以在任何条件下将第一指令集向协处理器迁移；

[0137] 举例说明，CPU在执行第一指令集的过程中一旦接收到指令集迁移指令，则将该第一指令集迁移至协处理器；其中，由以下三个条件中任一条件触发该指令集迁移指令：

[0138] 第一个条件，人为触发该指令集迁移指令，例如，CPU和协处理器同时集成在一数据处理设备中，人为操作数据处理设备触发该指令集迁移指令；

[0139] 第二个条件，CPU根据CPU使用率确定是否将第一指令集迁移至协处理器，如果CPU的CPU使用率大于第一阈值，触发该指令集迁移指令；

[0140] 第三个条件，CPU根据该CPU使用内存的内存使用率确定是否将第一指令集迁移至协处理器，如果CPU中该内存的内存使用率大于第二阈值，触发该指令集迁移指令。

[0141] 需说明的是，因第一指令集是根据表示CPU的指令集的二进制码对源代码编译得到，编译得到的第一指令集中每条二进制代码的操作码均属于表示CPU的指令集的二进制码，所以第一指令集中代表计算机指令的每条二进制码都能够被CPU识别和执行。通常，一条代表计算机指令的二进制码触发一个计算机操作，例如，X86处理器上运行的进程执行到一条二进制码“1101100011100000”，进行浮点数的减法运算这一计算机操作；再例如，X86处理器上运行的进程执行一条二进制码“1101100011100000”，进行将保存的FPU寄存器的状态恢复到FPU寄存器中这一计算机操作。

[0142] 需说明的是，第一指令集中的每条二进制代码不但包括操作码，可能还包括操作数；操作码是采用二进制码表示的，操作数也是采用二进制码表示的。

[0143] 本实施例中，CPU向协处理器迁移第一指令集，协处理器执行步骤A401接收所述CPU迁移的第一指令集。例如，CPU向协处理器迁移第一进程的过程中向协处理器发送第一指令集，所述协处理器执行步骤A401接收所述CPU迁移的第一指令集。

[0144] 步骤A402，所述协处理器根据所述第一指令集获得第二指令集，其中，所述第二指令集中的二进制代码用于指示所述协处理器在所述第二操作系统中执行所述计算机操作。

[0145] 具体地，按照上述将CPU的指令集划分为第一部分子指令集、第二部分子指令集和第三部分子指令集，协处理器的指令集不包含第三部分子指令集，因此协处理器的指令集与CPU的指令集存在不同；并且对于第二部分子指令集包含的计算机指令，中央处理器201支持的表示该计算机指令的二进制码，与协处理器202支持的表示该计算机指令的二进制码是不相同的；因此，表示第二部分子指令集包含的计算机指令的二进制码，和表示第三部分子指令集包含的计算机指令的二进制码，均不能够被协处理器识别和执行。鉴于此，本实施例提供步骤A402将第一指令集中的部分二进制代码或全部二进制代码转换，转换得到第二指令集；相对于第一指令集，第二指令集包含更多能够被协处理器识别和执行的二进制码，所述第二指令集具有更好的可识别性和可执行性。与第一指令集类似，第二指令集包含的每个二进制码分别触发计算机操作，第二指令集包含的一个二进制码触发一个计算机操作，并且本实施例期望协处理器执行第二指令集触发的计算机操作与CPU执行第一指令集触发的计算机操作相同。

[0146] 可选地，若所述第一指令集包含有表示计算机指令(该第一部分子指令集中的计算机指令)的二进制码，步骤A402在根据第一指令集获取第二指令集时，直接将所述第一指令集包含的该二进制码获取到第二指令集。

[0147] 可选地，若所述第一指令集包含有表示计算机指令(该第二部分子指令集中的计算机指令)的二进制码，步骤A402在根据第一指令集获取第二指令集时，将第一指令集包含的该二进制码转换为协处理器支持的表示该计算机指令的二进制码，将转换的二进制码获取到第二指令集。

[0148] 可选地，若所述第一指令集包含有表示计算机指令(该第三部分子指令集中的计算机指令)的二进制码，步骤A402在根据第一指令集获取第二指令集时，直接将第一指令集

包含的该二进制码获取到第二指令集。

[0149] 需说明的是，如果CPU向协处理器迁移第一进程，协处理器不但从CPU接收执行该第一进程所需的第一指令集，还从CPU获取其他与第一进程相关的数据，包括该第一进程的进程状态和该第一进程相关的寄存器值；该进程状态包括进程优先级、进程标识符、栈指针等进程运行的必要状态信息；协处理器将该第一进程相关的寄存器值存储至协处理器的寄存器中。另外，如果协处理器的寄存器与CPU的寄存器不属于同一种寄存器，步骤A402在根据第一指令集获取第二指令集时，还需将第一指令集中的二进制码中CPU的寄存器地址替换为协处理器的寄存器地址，将替换的协处理器的寄存器地址获取到第二指令集；可选地，如果协处理器的寄存器与CPU的寄存器属于同一种寄存器，步骤A402在根据第一指令集获取第二指令集时，直接将第一指令集中的二进制代码中CPU的寄存器地址获取到第二指令集。

[0150] 需说明的是，如果CPU向协处理器迁移第一线程，协处理器不但从CPU接收执行该第一线程所需的第一指令集，还从CPU获取其他与第一线程相关的数据，包括该第一线程的线程状态和该第一线程相关的寄存器值。协处理器将该第一线程相关的寄存器值存储至协处理器的寄存器中；与CPU向协处理器迁移的是执行第一线程所需的第一指令集类似，此处也根据协处理器的寄存器与CPU的寄存器是否不属于同一种寄存器，确定在步骤A402在根据第一指令集获取第二指令集时将CPU的寄存器地址还是将协处理器的寄存器地址获取到第二指令集。

[0151] 需说明的是，如果CPU向协处理器迁移的第一指令集是二进制代码的集合，协处理器还从CPU获取其他执行第一指令集所需的数据，包括该第一指令集相关的寄存器值；协处理器将该第一指令集相关的寄存器值存储至协处理器的寄存器中。与CPU向协处理器迁移的是执行第一线程所需的第一指令集类似，此处也根据协处理器的寄存器与CPU的寄存器是否不属于同一种寄存器，确定在步骤A402在根据第一指令集获取第二指令集时将CPU的寄存器地址还是将协处理器的寄存器地址获取到第二指令集。

[0152] 步骤A403，所述协处理器执行所述第二指令集中的二进制代码。

[0153] 如果CPU向协处理器迁移的是执行第一进程所需的第一指令集，第二操作系统在协处理器上，根据第一进程的进程状态和寄存器值确定第二进程的进程运行节点；从该进程运行节点开始，使用协处理器的寄存器，执行第二指令集中的二进制代码来运行第二进程。

[0154] 如果CPU向协处理器迁移的是执行第一线程所需的第一指令集，第二操作系统在协处理器上，根据第一线程的线程状态和寄存器值确定第二线程的线程运行节点；从该线程运行节点开始，使用协处理器的寄存器，执行第二指令集中的二进制代码来运行第二线程。

[0155] 如果CPU向协处理器迁移的第一指令集是二进制代码的集合，协处理器使用协处理器的寄存器，执行第二指令集。

[0156] 本实施例中，协处理器针对CPU迁移来的第一指令集，即使根据CPU的指令集编译得到的第一指令集包含协处理器不能识别的二进制代码，执行步骤A402能够部分或全部转换该二进制代码并对应生成第二指令集，协处理器能够识别第二指令集的识别率大于能够识别第一指令集的识别率，协处理器执行第二指令集，为CPU省去了运行第一指令集所需的

负荷。

[0157] 可选地，如果协处理器在步骤A403中完成执行第二指令集，根据具体应用场合确定是否将第二指令集的执行结果反馈CPU；如果是由CPU根据该结果执行其他计算机操作，协处理器将该结果反馈CPU；如果是由协处理器根据该结果执行其他计算机操作，协处理器可不将该结果反馈CPU；例如，对于所述第二指令集的执行结果，如果下一个计算机操作是由CPU控制显式模块显式该执行结果，则协处理器将该执行结果反馈CPU，如果协处理器能够直接控制显示模块，并且下一个计算机操作是由协处理器控制显式模块显式该执行结果，则协处理器可不需将该结果反馈CPU，直接控制显式模块显式该执行结果。

[0158] 可选地，针对第一指令集包含表示所述第二部分子指令集中的计算机指令的二进制码这一场景，对步骤A402做一细化，参见图5，所述协处理器根据所述第一指令集获得第二指令集包括：

[0159] 步骤A4021，所述协处理器在预先设置的翻译表中匹配所述第一指令集中的二进制代码的操作码，若所述第一指令集中的第一二进制代码的操作码在所述翻译表中被匹配到，则根据所述翻译表中所述第一二进制代码的操作码对应的匹配项，将所述第一二进制代码的操作码翻译为第二二进制代码的操作码，获得所述第二二进制代码，所述协处理器根据获得的至少一条所述第二二进制代码获得所述第二指令集，其中，所述翻译表包含相同的计算机指令分别编译生成的在所述第一操作系统和所述第二操作系统中不同操作码之间的对应关系，所述第二二进制代码为适用于所述第二操作系统的二进制代码。

[0160] 具体地，对于上述第二部分子指令集包含的每个计算机指令，协处理器支持的指令集也包含相同计算机指令，但是，CPU支持的表示该计算机指令的二进制码与协处理器支持的表示该计算机指令的二进制码不同。为让协处理器识别出该计算机指令，建立了翻译表，该翻译表针对第二部分子指令集中的每个计算机指令，分别记录每条计算机指令的对应关系；该计算机指令的对应关系包含：CPU支持的表示该计算机指令的二进制码，和协处理器支持的表示该计算机指令的二进制码。

[0161] 本实施例定义特定计算机指令为翻译表中记录有所述对应关系的计算机指令；因此，本实施例将第二部分子指令集包含的每个计算机指令的所述对应关系添加入翻译表，则第二部分子指令集包含的每个计算机指令均属于特定计算机指令。

[0162] 协处理器的第二操作系统加载翻译表后，对于步骤A401从CPU向协处理器迁移的第一指令集，步骤A4021根据翻译表遍历该第一指令集中的每条二进制代码，来匹配查找是否存在第一二进制代码，该第一二进制代码的操作码为该翻译表中记录的CPU支持的二进制码(即CPU支持的表示第二部分子指令集中的计算机指令的二进制码)。

[0163] 协处理器根据所述第一指令集获得第二指令集时，步骤A4021针对所述第一指令集中的每条第一二进制代码，替换该第一二进制代码中的操作码为翻译表记录的该操作码对应的匹配项，该匹配项为协处理器支持的表示特定计算机指令(该第一二进制代码中的操作码表示的特定计算机指令)的二进制码，将操作码替换所得的二进制代码作为第二二进制代码；在第二二进制代码中，该匹配项为该第二二进制代码的操作码；将第二二进制代码获取到第二指令集中。以此可知，协处理器根据所述第一指令集获得第二指令集时，第一指令集中的每条第一二进制代码都会转换为对应的第二二进制代码，将第二二进制代码获取到第二指令集。

[0164] 举例说明，对于异或指令(XOR)，CPU支持的表达该异或指令的二进制码根据比较的对象不同而有所不同；如果比较两个寄存器的值，该异或指令的二进制码表示为“0011001”，如果比较寄存器的值和内存的值，该异或指令的二进制码表示为“0011000”；但在协处理器中，异或指令(XOR)的二进制码统一表示为“0011000”，在翻译表中记录“0011001”与“0011000”的映射关系，协处理器根据翻译表从第一指令集的第一二进制码的操作码匹配到“0011001”，则将“0011000”作为第二二进制码的操作码，如果第一二进制码具有操作数，此处对如何根据第一二进制码的操作数生成第二二进制码的操作数不做限定。

[0165] 另外，协处理器根据所述第一指令集获得第二指令集时，根据翻译表遍历查找出的不属于第一二进制代码的其他二进制代码，采用哪种方式获取与其他二进制代码对应的二进制代码到第二指令集不做限定。

[0166] 进一步可选地，从步骤A402如何处理第一指令集包含表示所述第一部分子指令集和/或第三部分子指令集中的计算机指令的二进制码这一场景，对步骤A402做一细化，参见图5，所述协处理器根据所述第一指令集获得第二指令集还包括：

[0167] 步骤A4022，若所述第一指令集中的第三二进制代码的操作码在所述翻译表中未被匹配到，则所述协处理器将所述第三二进制代码做为所述第二指令集中的二进制代码。

[0168] 具体地，第三二进制代码属于步骤A4021根据翻译表从第一指令集中遍历查找出的不属于第一二进制代码的其他二进制代码。

[0169] 如果第一指令集中的某条二进制代码的操作码为表示所述第一部分子指令集中的计算机指令的二进制码，则该条二进制代码的操作码不会在所述翻译表匹配到；如果第一指令集中的某条二进制代码的操作码为表示所述第三部分子指令集中的计算机指令的二进制码，则该条二进制代码的操作码不会在所述翻译表匹配到；因此，所述第三二进制代码的操作码，可能是表示所述第一部分子指令集中的计算机指令的二进制码，或者可能是所述第三部分子指令集中的计算机指令的二进制码。

[0170] 协处理器根据所述第一指令集获得第二指令集时，步骤A4022针对所述第一指令集中不能在所述翻译表匹配到操作码的二进制代码(即第三二进制码)，将该第三二进制代码从第一指令集直接获取到第二指令集。

[0171] 可选地，步骤A402可以包括步骤A4021和/或步骤A4022，在执行步骤A402时是否会执行步骤A4021或者步骤A4022，根据具体实施场景确定；一种实施场景，如果第一指令集中不存在特定计算机指令，并且CPU和协处理器使用同一种寄存器执行进程，则步骤A402包括步骤A4022；如果第一指令集包含有特定计算机指令，则执行步骤A402包括步骤A4021，图5示出了在步骤A402时需执行步骤A4021和/或步骤A4022的示意图。

[0172] 进一步可选地，如果在步骤A402中需执行步骤A4021和步骤A4022，可先执行步骤A4021再执行步骤A4022，或者步骤A4021和步骤A4022并行执行。

[0173] 对步骤A4021和步骤A4022并行执行做一具体举例，在根据翻译表遍历第一指令集查找第一二进制代码时，每确定第一指令集中的一条二进制代码是否为第一二进制代码，便根据确定结果确定执行步骤A4021或者执行步骤A4022，具体地，如果确定结果为该条二进制代码为第一二进制代码，则执行步骤A4021将该第一二进制代码对应的第二二进制代码获取到第二指令集，如果确定结果为该条二进制代码为第三二进制代码，则执行步骤

A4022将该第三二进制代码直接获取到第二指令集。

[0174] 可选地，根据翻译表遍历第一指令集查找第一二进制代码的查找顺序为：第一指令集中每条二进制代码的执行顺序。

[0175] 可选地扩展，翻译表记录的对应关系(CPU支持的表示计算机指令的二进制码，和协处理器支持的表示该计算机指令的二进制码)可以是一条或多条，步骤A4021使用的上述翻译表记录了与第二部分子指令集中的每个计算机指令匹配的该对应关系，但此处的翻译表可以少于步骤A4021使用的翻译表中记录的对应关系的条数，因此此处的翻译表是可以更新的，如向翻译表添加与第二部分子指令集中的某个计算机指令匹配的该对应关系，删除翻译表中的一条或者多条该对应关系；从而提供一种可替换的步骤A4021，即在执行步骤A4021时使用此处的翻译表替换使用上述的翻译表。

[0176] 可选地，针对CPU具有的寄存器与协处理器具有的寄存器不是同一种寄存器这一场景，对计算机指令处理方法做一可选细化，参见图6，在所述协处理器执行所述第二指令集中的二进制代码之前，所述方法还包括步骤A601；

[0177] 步骤A601，所述协处理器将所述第二指令集包含的二进制代码中所述CPU的寄存器地址转换为所述协处理器的寄存器地址。

[0178] 具体地，所述CPU的寄存器地址采用二进制码表示，所述协处理器的寄存器地址采用二进制码表示；CPU具有的寄存器与协处理器具有的寄存器不是同一种寄存器，表示CPU的寄存器地址的二进制码与表示协处理器的寄存器地址的二进制码不相同。

[0179] 协处理器为使用自己的寄存器运行CPU向协处理器迁移的第二指令集，在执行第二指令集之前，查找第二指令集的二进制代码中是否包含所述CPU的寄存器地址，如果查找到，根据匹配替换关系替换第二指令集中查找到的所述CPU的寄存器地址为对应的协处理器的寄存器地址，该匹配替换关系为协处理器的寄存器地址与所述CPU的寄存器地址的映射关系。

[0180] 可选地，作为步骤A601的替代方案，所述协处理器在步骤A402中根据所述第一指令集获得第二指令集时，便执行步骤A602将所述第一指令集包含的二进制代码中所述CPU的寄存器地址转换为所述协处理器的寄存器地址，将所述协处理器的寄存器地址获取到第二指令集中；这时，对步骤A602的替代方案与步骤A4021的执行顺序不做限定，通常，该替代方案与步骤A4021并行执行。

[0181] 可选地，所述CPU的寄存器为128位的寄存器或者为256位的寄存器，所述协处理器的寄存器为512位的寄存器。

[0182] 更进一步可选地，所述CPU的寄存器为128位的XMM寄存器或者为256位的YMM寄存器，所述协处理器的寄存器为256位的ZMM寄存器。无论CPU中的寄存器还是协处理器中的寄存器，在执行第二指令集时都是用于暂存计算机指令、数据和地址的，相对于基于XMM寄存器或基于YMM寄存器执行第二指令集，基于ZMM寄存器执行第二指令集，能够提高第二指令集的执行速度，提前完成第二指令集的执行。

[0183] 进一步可选地，第一指令集中使用CPU的寄存器的计算机指令属于矢量化指令，第二指令集中使用协处理器的寄存器的计算机指令属于矢量化指令。在步骤A402中根据所述第一指令集获得第二指令集时，根据第一指令集中使用CPU的寄存器的矢量化指令，对应地将使用协处理器的寄存器的矢量化指令获取到第二指令集中。

[0184] 可选地，所述第一指令集是所述CPU在所述CPU的CPU使用率大于第一阈值时向所述协处理器迁移的。

[0185] 具体地，CPU执行第一指令集，可选地还可以并行执行一条或多条其他二进制代码。如果所述CPU使用率大于第一阈值，代表CPU使用率过高，将第一指令集迁移至协处理器，减小CPU的负荷。

[0186] 下面以第一进程为例讲解如何筛选向协处理器迁移的第一指令集，当然，筛选第一进程的方式也适合筛选第一线程；筛选第一进程的方式详述如下：如果CPU仅运行一个进程，则该个进程为第一进程。如果CPU并行运行多个进程，对如何从CPU执行的多个进程中确定第一进程提供几种可选细化实现方式：

[0187] 第一种可选细化实现方式，在CPU的CPU使用率大于第一阈值的当前，从优先级小于优先级阈值的进程选取一个或多个第一进程，优选地，选取优先级最低的进程作为第一进程；

[0188] 第二种可选细化实现方式，在CPU的CPU使用率大于第一阈值的当前，从CPU占用率大于占用率阈值的进程选取一个或多个第一进程，优选地，选取CPU占用率最高的进程作为第一进程。

[0189] 进一步可选地，参见图7，步骤A401所述协处理器接收所述CPU迁移的第一指令集，包括步骤A4011和步骤A4012。

[0190] 步骤A4011，所述协处理器接收所述CPU发送的要迁移的所述第一指令集的地址，所述第一指令集的地址是指所述第一指令集在所述CPU的内存中存储的地址，其中，所述第一指令集的地址由所述CPU在所述CPU的内存使用率小于或等于第二阈值时向所述协处理器发送；

[0191] 步骤A4012，所述协处理器基于所述第一指令集的地址访问所述CPU的内存来获取所述第一指令集。

[0192] 具体地，CPU的CPU使用率大于第一阈值、CPU使用内存的内存使用率小于或等于第二阈值，代表CPU使用率过高但CPU使用内存的内存使用率没有过高；这种情况下，CPU向协处理器发送的是所述第一指令集的地址，可选地，所述第一指令集的地址为在CPU的内存中存储所述第一指令集的物理地址。协处理器在步骤A4011接收到所述第一指令集的地址后，执行步骤A4012根据所述第一指令集的地址访问所述CPU的内存，从所述CPU的内存读取第一指令集，再执行步骤A402根据第一指令集获取第二指令集，并将获取的第二指令集存储至CPU的内存中，一种可选的具体方式是，以获取的第二指令集替换CPU的内存中的第一指令集。

[0193] 一种具体实现步骤A4011和步骤A4012的可选方式是，CPU在CPU使用率大于第一阈值、且CPU使用内存的内存使用率小于或等于第二阈值时，将与第一指令集相关的数据在CPU的内存中的存储地址发送至协处理器；而后，协处理器经总线(如PCI-E总线)根据该存储地址访问CPU的内存，从CPU的内存读取与第一指令集相关的数据，从与第一指令集相关的数据提取第一指令集，再执行步骤A402根据第一指令集获取第二指令集，并将获取的第二指令集存储至CPU的内存中，使用CPU的内存执行第二指令集。

[0194] 其中，与第一指令集相关的数据包括第一指令集，还包括第一指令集的运行状态(如第一进程的进程状态)等其他执行第一指令集所需的数据。

[0195] 可选地，所述第一指令集由所述CPU在所述CPU的内存使用率大于第二阈值时向所述协处理器发送。

[0196] 具体地，如果CPU使用内存的内存使用率大于第二阈值，代表CPU使用内存的内存使用率过高；这种情况下，无论CPU的CPU使用率是否大于第一阈值，CPU在均向协处理器迁移第一指令集，步骤A401接收该第一指令集并存储至协处理器的内存中。

[0197] 一种具体的可选实现方式是，在CPU使用内存的内存使用率大于第二阈值时，CPU从其使用的内存中读取与第一指令集相关的数据，将读取的该与第一指令集相关的数据发送至协处理器；协处理器接收与第一指令集相关的数据，在协处理器的内存存储该与第一指令集相关的数据；继而协处理器再从与第一指令集相关的数据提取第一指令集，执行步骤A402根据第一指令集获取第二指令集，并将获取的第二指令集存储至协处理器的内存中，使用协处理器的内存执行第二指令集。

[0198] 可选地，如果步骤A403执行第二指令集中出现异常，可采用以下四种可选方式中的任一种可选方式处理。

[0199] 第一种可选方式，参见图8，步骤A403中所述协处理器执行所述第二指令集中的二进制代码，具体包括步骤A801、步骤A802、步骤A803和步骤A804。

[0200] 步骤A801，所述协处理器依次执行所述第二指令集中的二进制代码；

[0201] 步骤A802，如果检测到执行所述第二指令集时出现二进制代码识别异常，则确定触发异常的第四二进制代码；

[0202] 步骤A803，将所述第四二进制代码转换为中间代码，再将所述中间代码转换为适用于所述第二操作系统的第五二进制代码；其中，第五二进制代码为一条或多条二进制代码；

[0203] 步骤A804，执行所述第五二进制代码，并继续执行所述第二指令集中所述第四二进制代码之后的二进制代码。

[0204] 具体地，步骤A402根据第一指令集获取第二指令集，虽然获取到的第二指令集中的二进制代码用于指示所述协处理器在所述第二操作系统中执行计算机操作，但第二指令集中的每条二进制代码不一定都能够被协处理器识别，对于协处理器执行到不能识别的二进制代码会触发二进制代码识别异常，本实施例定义触发二进制代码异常的二进制指令为第四二进制代码。

[0205] 一种第二指令集中的二进制代码不能够被识别的场景是，如果在步骤A402中将第一指令集中表示第三部分子指令集中每个计算机指令的二进制码直接获取到第二指令集中，则协处理器在运行第二指令集时执行到包含以该二进制码为操作码的二进制代码，无法识别该条二进制码而触发二进制代码识别异常，因此，该条二进制代码属于第四二进制代码。

[0206] 所述协处理器步骤A801依次执行所述第二指令集中的二进制代码的过程中，若步骤A802检测到二进制代码识别异常，确定触发该二进制代码识别异常的第四二进制代码；继而步骤A803将所述第四二进制代码转换为中间代码，再将所述中间代码转换为协处理器能够识别的第五二进制代码；继而，步骤A804执行所述第五二进制代码，再继续执行所述第二指令集中所述第四二进制代码之后的二进制代码。其中，该中间代码与第四二进制代码具有映射关系，可以是一条或多条中间代码对应一条第四二进制代码的映射关系；同时，该

中间代码还与第五二进制代码具有映射关系;在满足中间代码同时与第四二进制代码和第五二进制代码均具有映射关系的条件下,对中间代码的具体表现形式不做限定;举例说明,采用Java字节码作为中间代码,在将第四二进制代码转换为相应的Java字节码时确定该第四二进制代码与相应Java字节码的映射关系,在将相应Java字节码转换为第五二进制代码为时确定该相应Java字节码与第五二进制代码的映射关系;类似地,在步骤A803将所述第四二进制代码转换为第五二进制代码的过程中,还可选用simics模拟器的中间代码实现,还可选用qemu模拟器的中间代码实现。

[0207] 举例说明,MIC对于上述第三部分子指令集中包括的CMOV、OUT、PAUSE、SYSEXIT等22种计算机指令是不支持的,MIC执行到以表示该计算机指令的二进制码为操作码的第四二进制代码将无法识别,会出现二进制代码识别异常;以条件跳转指令(CMOV)为例,MIC执行到以表示该条件跳转指令的二进制码为操作码的第四二进制代码会触发二进制代码识别异常,在步骤A802检测到该二进制代码识别异常,在步骤A803根据该条件跳转指令确定中间代码表示的条件判断指令和移动指令,再将该中间代码表示的条件判断指令和移动指令分别翻译成MIC可识别的条件判断指令和移动指令(MOV),MIC在步骤A804先执行条件判断指令确定是否满足跳转条件,如果满足,执行移动指令(MOV),再执行第二指令集中第四二进制代码后的二进制代码;需说明的是,此处是根据条件跳转指令(CMOV)所带的操作数确定中间代码表示的条件判断指令和移动指令分别所带的操作数,再根据中间代码表示的条件判断指令和移动指令分别带的操作数确定MIC可识别的条件判断指令和移动指令(MOV)分别带的操作数;按照上述对22种计算机指令的分类,对于第一类和第三类包含的计算机指令,步骤A803能够将以表示该计算机指令的二进制码为操作码的第四二进制代码经中间代码转换为第五二进制代码,根据一条第四二进制代码转换出的第五二进制代码可以是一条或多条,此处对第五二进制代码的条数不做限定。

[0208] 可选地,步骤A403执行第二指令集出现异常,暂停执行第二指令集,并输出异常信息,所述异常信息包括触发异常的第四二进制代码、异常类型、异常执行结果等等;在可选的具体实施中,实时将执行第二指令集的状态信息等写入第二指令集的运行日志中,包括将执行第二指令集出现异常的异常信息也会写入第二指令集的运行日志中。如果根据异常信息确定异常为二进制代码识别异常,则步骤A802根据异常信息确定触发异常的第四二进制代码。

[0209] 在第一中可选方式中,即使协处理器执行第二指令集出现二进制代码识别异常,能够将该第四二进制代码经中间代码间接转换为协处理器支持的第五二进制代码,从第五二进制代码继续执行第二指令集,有效克服该异常并保证第二指令集的正常执行;以此类推,对于在执行第二指令集的过程中每次出现的异常,都能有效克服,在每次克服异常后继续执行第二指令集。

[0210] 可选地,通常为优化协处理器的执行效率,针对协处理器的特殊应用开发了仅协处理器支持的扩展指令集,该扩展指令集包括的计算机指令(采用二进制码表示)仅协处理器支持但CPU不支持;预先确定中间代码表示的操作码与协处理器支持的指令集中一个或多个计算机指令之间的映射关系,可能会确定所述中间代码表示的操作码与所述扩展指令集包括的一个或多个计算机指令之间的映射关系,因此对于步骤A803将中间代码转换为的第五二进制代码,该第五二进制代码的操作码可能是表示该扩展指令集中的计算机指令的

二进制码。这样，将第四二进制代码间接地转换为以表示该扩展指令集包括的计算机指令的二进制码为操作码的二进制代码之后，不但能够解决该异常信息所指的异常，还能够提高协处理器指定第二指令集的效率。

[0211] 第二种可选方式，参见图9，在步骤A803将所述第四二进制代码转换为中间代码之前，包括步骤A901和步骤A902。

[0212] 步骤A901，向所述CPU发送指令集回迁请求；

[0213] 步骤A902，接收所述CPU发送的拒绝回迁指令。

[0214] 具体地，步骤A403执行第二指令集出现二进制代码识别异常，执行步骤A901向所述CPU发送指令集回迁请求；CPU响应该指令集回迁请求，并确定是否将第二指令集回迁CPU执行，如果确定不将第二指令集回迁，则向协处理器反馈拒绝回迁指令；协处理器在步骤A902中接收接收到该拒绝回迁指令，则执行步骤A803将所述第四二进制代码转换为中间代码。

[0215] CPU响应该指令集回迁请求的一种可选方式是，根据CPU的负荷确定是否将第二指令集回迁CPU执行；该CPU的负荷包括：CPU使用率，和CPU使用内存的内存使用率。如果所述CPU的CPU使用率大于所述第三阈值，或者如果所述CPU使用内存的内存使用率大于第四阈值，则向所述协处理器发送拒绝回迁指令。

[0216] 第三种可选方式，参见图10，步骤A403中所述协处理器执行所述第二指令集中的二进制代码，包括步骤B1001、步骤B1002和步骤B1003。

[0217] 步骤B1001，所述协处理器依次执行所述第二指令集中的二进制代码；

[0218] 步骤B1002，如果检测到执行所述第二指令集时出现二进制代码识别异常，则确定触发异常的第六二进制代码；

[0219] 步骤B1003，根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集，并向所述CPU迁移所述第三指令集。

[0220] 具体地，步骤A402根据第一指令集获取第二指令集，第二指令集中的每条二进制代码不一定都能够被协处理器识别，对于协处理器执行到不能识别的二进制代码会触发二进制代码识别异常，本实施例定义触发二进制代码异常的二进制指令为第六二进制代码，定义第六二进制代码与定义第四二进制代码同原理，可参见第一种可选方式中对定义第四二进制代码的相关解释。同原理地，如果在步骤A402中将第一指令集中表示第三部分子指令集中每个计算机指令的二进制码直接获取到第二指令集中，则协处理器在运行第二指令集时执行到以该二进制码为操作码的二进制代码，无法识别该二进制代码而触发二进制代码识别异常，因此，该二进制代码属于第六二进制代码。

[0221] 与第一种可选方式不同的是，所述协处理器在步骤B1001依次执行所述第二指令集中的二进制代码的过程中，若步骤B1002检测到二进制代码识别异常并确定触发该二进制代码识别异常的第六二进制代码后，执行步骤B1003来处理该二进制代码识别异常。

[0222] 协处理器在步骤B1003根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集，与步骤A402根据所述第一指令集获得第二指令集的实现原理相同，可参照上述对步骤A402的相关解释，及对步骤A402的可选细化的相关解释，例如对步骤A4021、步骤A4022等的相关解释。与步骤A4021对应地，步骤B1003根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作

系统的第三指令集时，根据翻译表遍历所述第六二进制代码开始的第二指令集中的每条二进制代码，匹配查找是否存在第七二进制代码，该第七二进制代码的操作码为该翻译表中记录的协处理器支持的二进制码，替换该第七二进制代码中的操作码为翻译表记录的该操作码对应的匹配项，该匹配项为CPU支持的表示特定计算机指令(该第七二进制代码中的操作码表示的特定计算机指令)的二进制码，将操作码替换后的二进制代码作为第八二进制代码；在第八二进制代码中，该匹配项为该第八二进制代码的操作码；将第八二进制代码获取到第三指令集中。以此可知，步骤B1003根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集时，所述第六二进制代码开始的第二指令集中的每条第七二进制代码都会转换为对应的第八二进制代码，将第八二进制代码获取到第三指令集。

[0223] 另可选地，如果协处理器的寄存器与CPU的寄存器不是同一种寄存器，有两种处理方式：

[0224] 第一种处理方式，由协处理器处理；具体地，所述协处理器在步骤B1003根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集时，便执行步骤B1003将第二指令集中所述第六二进制代码开始的二进制代码包含的所述协处理器的寄存器地址转换为所述CPU的寄存器地址，将所述CPU的寄存器地址获取到第三指令集中；这时，对根据翻译表获取第八二进制代码和第一种处理方式转换寄存器地址的执行顺序不做限定，通常并行执行；

[0225] 第二种处理方式，由CPU处理；具体地，CPU接收到协处理器在步骤B1003迁移的第三指令集后，将第三指令集中所述协处理器的寄存器地址替换为所述CPU的寄存器地址。

[0226] 可选地，如果第二指令集存储在CPU的内存中，则步骤B1003根据第二指令集在CPU的内存中的存储地址，以第三指令集替换CPU的内存中存储的第二指令集。如果第二指令集存储在协处理器的内存中，则步骤B1003将第三指令集向CPU迁移，使得CPU将第三指令集存储在CPU的内存中。

[0227] 本可选方式中，协处理器执行第六二进制代码触发二进制代码识别异常，则向CPU迁移未执行完的第二指令集；向CPU迁移未执行完的第二指令集的过程中，如果是由协处理器的内存中存储的与该未执行完的第二指令集相关的数据，将协处理器的内存中存储的与该未执行完的第二指令集相关的数据向CPU发送，与该未执行完的第二指令集相关的数据包括：向CPU迁移的第三指令集，以便CPU将该未执行完的第二指令集相关的数据存储至CPU的内存中。另外，协处理器还将与该未执行完的第二指令集相关的寄存器值向CPU发送，CPU将与该未执行完的第二指令集相关的寄存器值存储至CPU的寄存器(与第三指令集中的寄存器地址对应)中。

[0228] 第四种可选方式，参见图11，在步骤B1003根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集之前，还包括步骤B1101和步骤B1102。

[0229] 步骤B1101，向所述CPU发送指令集回迁请求；

[0230] 步骤B1102，接收所述CPU发送的指令集回迁响应。

[0231] 具体地，步骤A403执行第二指令集出现二进制代码识别异常，执行步骤A1101向所述CPU发送指令集回迁请求；CPU响应该指令集回迁请求，并确定是否将第二指令集回迁CPU

执行，如果确定将第二指令集回迁，则向协处理器反馈指令集回迁响应；协处理器在步骤B1003中接收接收到该指令集回迁响应，则执行步骤B1003根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集。

[0232] CPU响应该指令集回迁请求的一种可选方式是，根据CPU的负荷确定是否将第二指令集回迁CPU执行；该CPU的负荷包括：CPU使用率，和CPU使用内存的内存使用率。如果所述CPU的CPU使用率小于或等于第五阈值、且所述CPU使用内存的内存使用率小于或等于第六阈值，则向所述协处理器发送指令集回迁响应。

[0233] 本发明一实施例，图12是本实施例的协处理器202的一种可选逻辑结构示意图；所述协处理器202应用于处理器系统，所述处理器系统包括所述协处理器202和运行第一操作系统的中央处理器(CPU)，所述协处理器202上运行第二操作系统；

[0234] 所述协处理器202包括：

[0235] 第一指令集接收单元2021，用于接收所述中央处理器迁移的第一指令集，所述第一指令集用于指示所述中央处理器在所述第一操作系统中执行计算机操作，所述第一指令集为适用于所述第一操作系统的二进制代码的集合；

[0236] 第二指令集获得单元2022，用于根据所述第一指令集获得第二指令集，其中，所述第二指令集中的二进制代码用于指示所述协处理器202在所述第二操作系统中执行所述计算机操作；

[0237] 第二指令集执行单元2023，用于执行所述第二指令集中的二进制代码。

[0238] 可选地，所述第二指令集获得单元2022，用于根据所述第一指令集获得第二指令集，包括：

[0239] 所述第二指令集获得单元2022，用于在预先设置的翻译表中匹配所述第一指令集中的二进制代码的操作码，若所述第一指令集中的第一二进制代码的操作码在所述翻译表中被匹配到，则根据所述翻译表中所述第一二进制代码的操作码对应的匹配项，将所述第一二进制代码的操作码翻译为第二二进制代码的操作码，获得所述第二二进制代码，根据获得的至少一条所述第二二进制代码获得所述第二指令集，其中，所述翻译表包含相同的计算机指令分别编译生成的在所述第一操作系统和所述第二操作系统中不同的操作码之间的对应关系，所述第二二进制代码为适用于所述第二操作系统的二进制代码。

[0240] 可选地，参见图13，所述协处理器202还包括：

[0241] 寄存器地址转换单元2024，用于将所述第二指令集包含的二进制代码中所述中央处理器的寄存器地址转换为所述协处理器202的寄存器地址。

[0242] 可选地，所述第二指令集获得单元2022，还用于若所述第一指令集中的第三二进制代码的操作码在所述翻译表中未被匹配到，则所述协处理器202将所述第三二进制代码做为所述第二指令集中的二进制代码。

[0243] 可选地，所述第一指令集是所述中央处理器在所述中央处理器的CPU使用率大于第一阈值时向所述协处理器202迁移的。

[0244] 可选地，所述第一指令集接收单元2021，用于接收所述中央处理器迁移的第一指令集，包括：

[0245] 所述第一指令集接收单元2021，用于接收所述中央处理器发送的要迁移的所述第一指令集的地址，并基于所述第一指令集的地址访问所述中央处理器的内存来获取所述第

一指令集;其中,所述第一指令集的地址是指所述第一指令集在所述中央处理器的内存中存储的地址,其中,所述第一指令集的地址由所述中央处理器在所述中央处理器的内存使用率小于或等于第二阈值时向所述协处理器202发送。

[0246] 可选地,所述第一指令集由所述中央处理器在所述中央处理器的内存使用率大于第二阈值时向所述协处理器202发送。

[0247] 可选地,所述第二指令集执行单元2023,用于执行所述第二指令集中的二进制代码,包括:

[0248] 所述第二指令集执行单元2023,用于依次执行所述第二指令集中的二进制代码;如果检测到执行所述第二指令集时出现二进制代码识别异常,则确定触发异常的第四二进制代码,将所述第四二进制代码转换为中间代码,再将所述中间代码转换为适用于所述第二操作系统的第五二进制代码,执行所述第五二进制代码,并继续执行所述第二指令集中所述第四二进制代码之后的二进制代码。

[0249] 可选地,所述第二指令集执行单元2023,还用于在将所述第四二进制代码转换为中间代码之前,向所述中央处理器发送指令集回迁请求,接收所述中央处理器发送的拒绝回迁指令。

[0250] 可选地,所述第二指令集执行单元2023,用于执行所述第二指令集中的二进制代码,包括:

[0251] 所述第二指令集执行单元2023,用于依次执行所述第二指令集中的二进制代码;如果检测到执行所述第二指令集时出现二进制代码识别异常,则确定触发异常的第六二进制代码;根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集,并向所述中央处理器迁移所述第三指令集。

[0252] 可选地,所述第二指令集执行单元2023,还用于根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集之前,向所述中央处理器发送指令集回迁请求,接收所述中央处理器发送的指令集回迁响应。

[0253] 本发明一实施例,图14是本实施例提供的协处理器1401的硬件结构示意图,示出了所述协处理器1401的一种硬件结构。

[0254] 如图14所示,协处理器1401与存储器1402通过总线1403连接,所述存储器1402用于存储计算机执行指令,所述协处理器1401读取所述存储器1402存储的所述计算机执行指令,执行上述实施例提供的计算机指令处理方法。该计算机指令处理方法的具体实现,参见上述实施例对该计算机指令处理方法的相关描述,在此不再赘述。

[0255] 其中,协处理器1401可以采用因特尔集成众核架构(Many Integrated Core,简称MIC),微处理器,应用专用集成电路(Application Specific Integrated Circuit,ASIC),或者一个或多个集成电路,用于执行相关程序,以实现上述方法实施例所提供的技术方案,包括执行上述实施例提供的计算机指令处理方法。

[0256] 其中,存储器1402可以是只读存储器(Read Only Memory,ROM),静态存储设备,动态存储设备或者随机存取存储器(Random Access Memory,RAM)。存储器1402可以存储操作系统和其他应用程序。在通过软件或者固件来实现上述方法实施例提供的技术方案时,用于实现上述方法实施例提供的技术方案的程序代码保存在存储器1402中,包括将应用于所述协处理器1401的上述实施例提供的计算机指令处理方法的程序代码保存在存储器1402

中，并由协处理器1401来执行。

[0257] 其中，总线1403可包括一通路，用于在所述协处理器1401中各个部件与存储器1402之间传送信息。

[0258] 应注意，尽管图14所示的所述协处理器1401仅仅示出了协处理器1401、存储器1402以及总线1403，但是在具体实现过程中，本领域的技术人员应当明白，所述协处理器1401还包含实现正常运行所必须的其他器件，例如通信接口。同时，根据具体需要，本领域的技术人员应当明白，所述协处理器1401还可包含实现其他附加功能的硬件器件。此外，本领域的技术人员应当明白，所述协处理器1401也可仅仅包含实现上述方法实施例所必须的器件，而不必包含图14中所示的全部器件。

[0259] 本发明一实施例，提供一种系统200，参见图1，所述处理器系统200包括中央处理器201(CPU)和协处理器202，所述CPU上运行第一操作系统，所述协处理器202上运行第二操作系统；

[0260] 所述中央处理器201，用于向协处理器202迁移第一指令集；

[0261] 所述协处理器202，用于执行上述实施例或者及上述实施例的可选细化方式提供的计算机指令处理方法。

[0262] 在本申请所提供的几个实施例中，应该理解到，所揭露的系统，设备和方法，可以通过其它的方式实现。例如，以上所描述的设备实施例仅仅是示意性的，例如，所述模块和单元的划分，仅仅为一种逻辑功能划分，实现时可以有另外的划分方式，例如多个模块或单元或组件可以结合或者可以集成到另一个系统，或一些特征可以忽略，或不执行。另一点，所显示或讨论的相互之间的耦合或直接耦合或通信连接可以是通过一些接口，设备或模块的间接耦合或通信连接，可以是电性，机械或其它的形式。

[0263] 所述作为分离部件说明的模块可以是或者也可以不是物理上分开的，作为模块的部件可以是或者也可以不是物理模块，即可以位于一个地方，或者也可以分布到多个网络模块上。可以根据实际的需要选择其中的部分或者全部模块来实现本实施例方案的目的。

[0264] 另外，在本发明各个实施例中的各功能模块可以集成在一个处理模块中，也可以是各个模块单独物理存在，也可以两个或两个以上模块集成在一个模块中。上述集成的模块既可以采用硬件的形式实现，也可以采用硬件加软件功能模块的形式实现。

[0265] 上述以软件功能模块的形式实现集成的模块，可以存储在一个计算机可读取存储介质中。上述软件功能模块存储在一个存储介质中，包括若干指令用以使得一台计算机设备(可以是个人计算机，服务器，或者网络设备等)执行本发明各个实施例所述方法的部分步骤。而前述的存储介质包括：移动硬盘、只读存储器(英文：Read-Only Memory，简称ROM)、随机存取存储器(英文：Random Access Memory，简称RAM)、磁碟或者光盘等各种可以存储程序代码的介质。

[0266] 总之，以上所述仅为本发明技术方案的较佳实施例而已，并非用于限定本发明的保护范围。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。

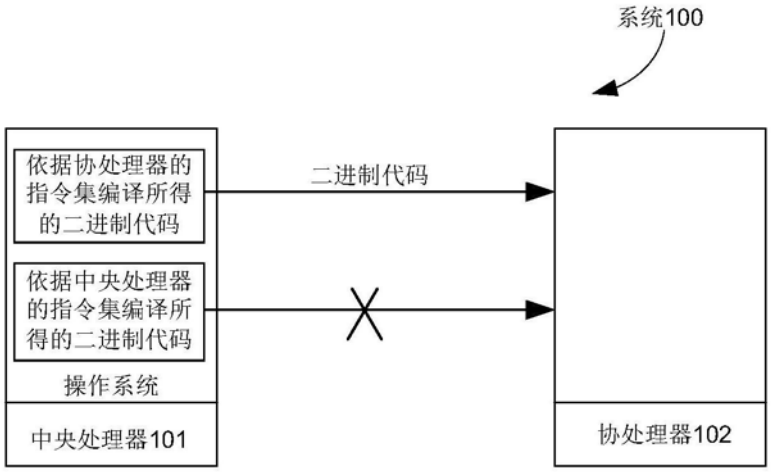

系统100
依据协处理器的指令集编译所得的二进制代码
依据中央处理器的指令集编译所得的二进制代码
二进制代码
操作系统
中央处理器101
协处理器102

图1

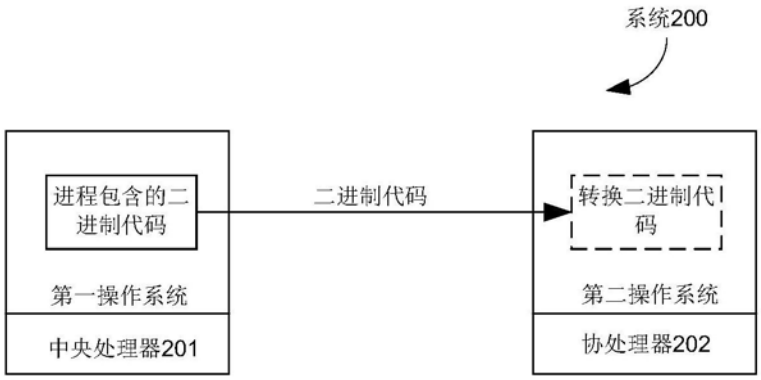

系统200
进程包含的二进制代码
二进制代码
转换二进制代码
第一操作系统
第二操作系统
中央处理器201
协处理器202

图2

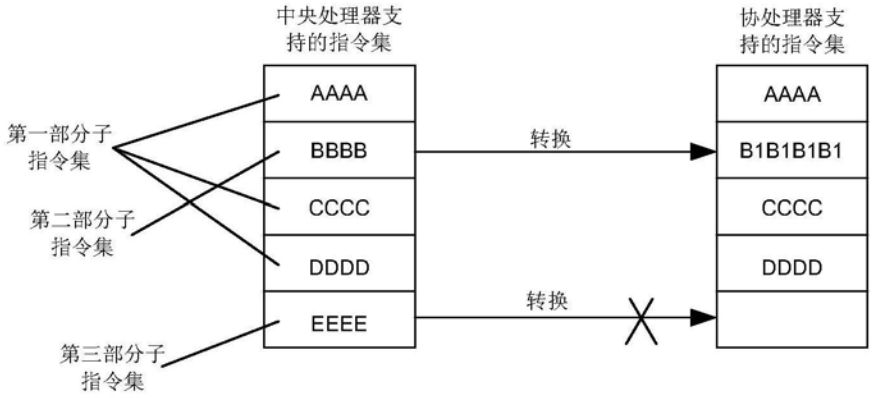

中央处理器支持的指令集
协处理器支持的指令集
AAAA
BBBB
CCCC
DDDD
EEEE
转换
转换
AAAA
B1B1B1B1
CCCC
DDDD
第一部分子指令集
第二部分子指令集
第三部分子指令集

图3

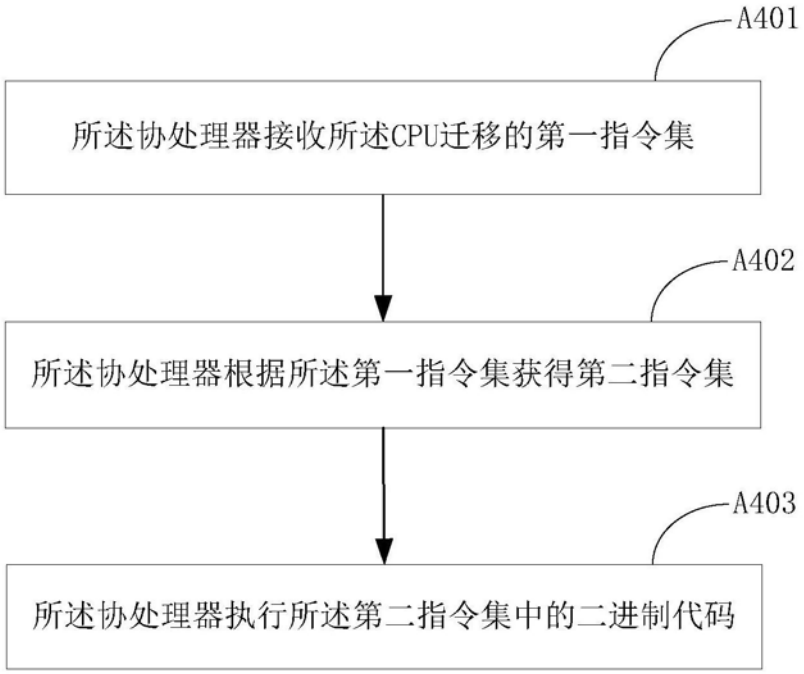

A401
所述协处理器接收所述CPU迁移的第一指令集
A402
所述协处理器根据所述第一指令集获得第二指令集
A403
所述协处理器执行所述第二指令集中的二进制代码

图4

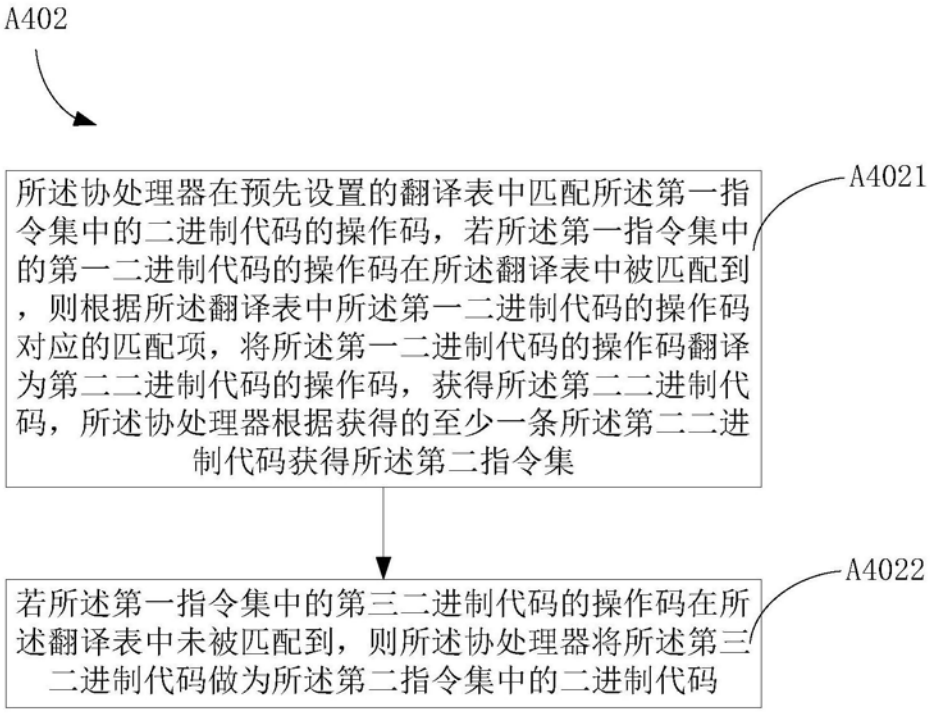

A402
所述协处理器在预先设置的翻译表中匹配所述第一指令集中的二进制代码的操作码，若所述第一指令集中的第一二进制代码的操作码在所述翻译表中被匹配到，则根据所述翻译表中所述第一二进制代码的操作码对应的匹配项，将所述第一二进制代码的操作码翻译为第二二进制代码的操作码，获得所述第二二进制代码，所述协处理器根据获得的至少一条所述第二二进制代码获得所述第二指令集
A4021
若所述第一指令集中的第三二进制代码的操作码在所述翻译表中未被匹配到，则所述协处理器将所述第三二进制代码做为所述第二指令集中的二进制代码
A4022

图5

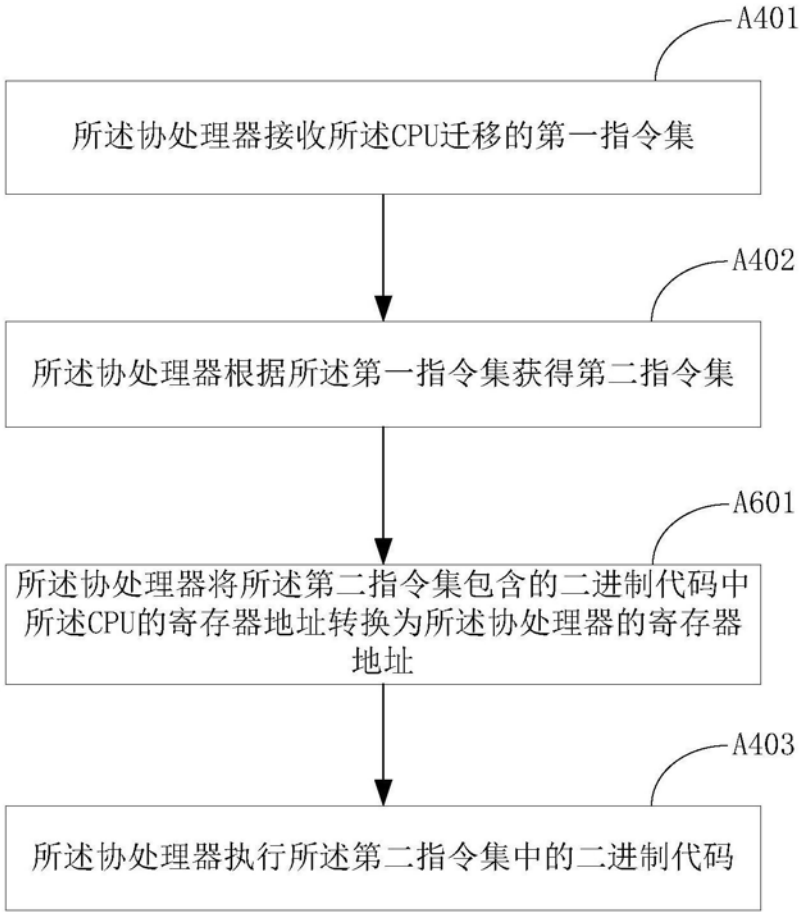

A401
所述协处理器接收所述CPU迁移的第一指令集
A402
所述协处理器根据所述第一指令集获得第二指令集
A601
所述协处理器将所述第二指令集包含的二进制代码中所述CPU的寄存器地址转换为所述协处理器的寄存器地址
A403
所述协处理器执行所述第二指令集中的二进制代码

图6

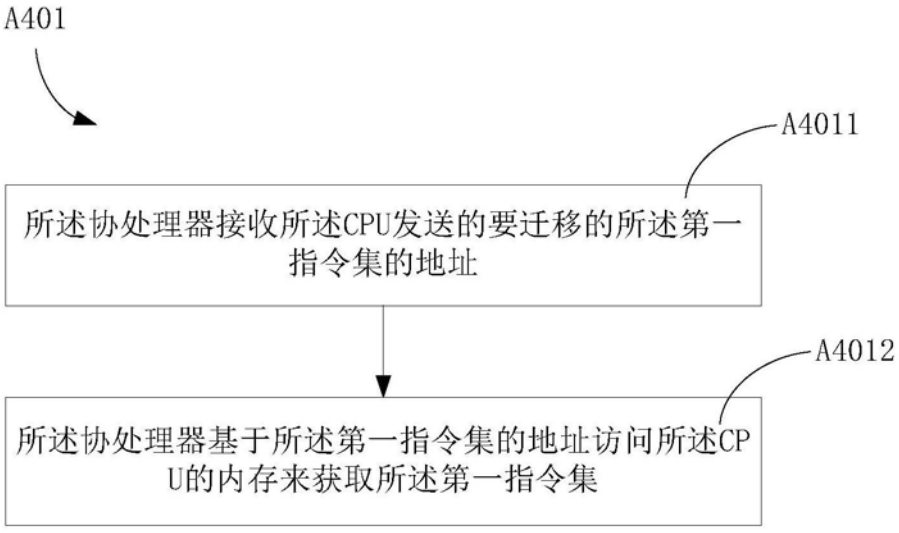

A401
A4011
所述协处理器接收所述CPU发送的要迁移的所述第一指令集的地址
A4012
所述协处理器基于所述第一指令集的地址访问所述CPU的内存来获取所述第一指令集

图7

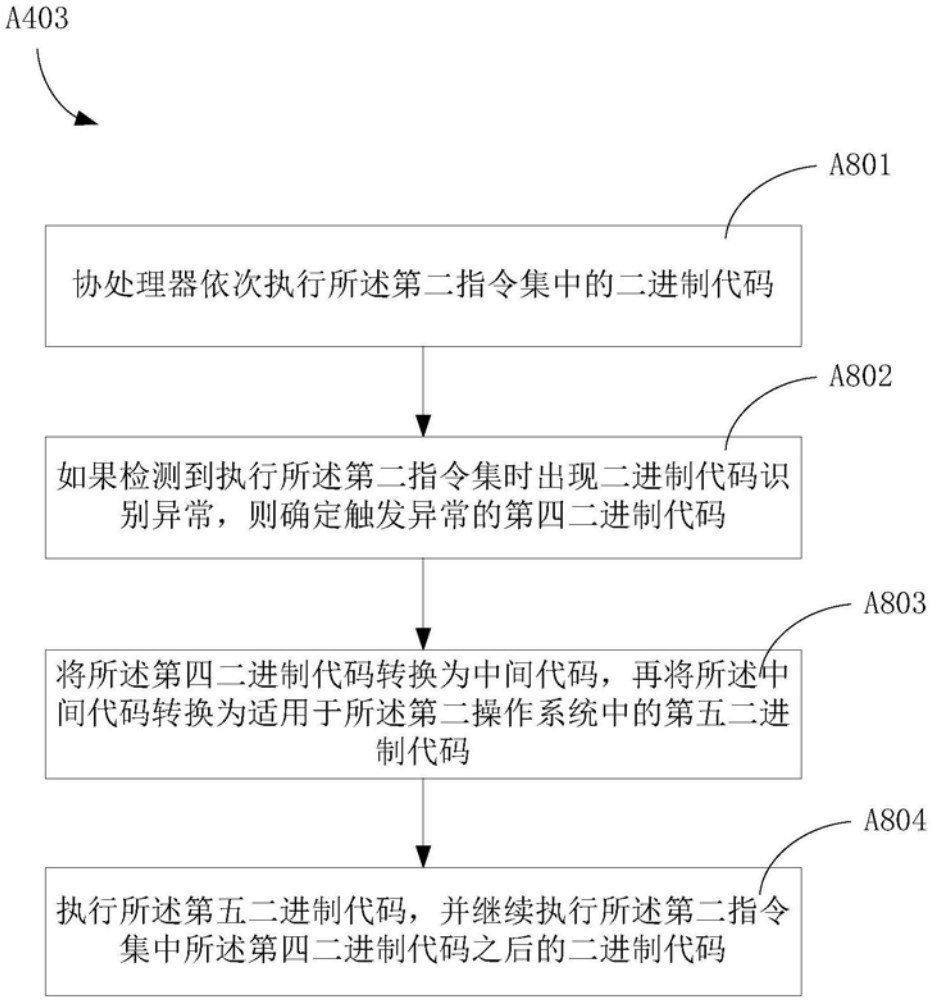

A403
A801
协处理器依次执行所述第二指令集中的二进制代码
A802
如果检测到执行所述第二指令集时出现二进制代码识别异常，则确定触发异常的第四二进制代码
A803
将所述第四二进制代码转换为中间代码，再将所述中间代码转换为适用于所述第二操作系统中的第五二进制代码
A804
执行所述第五二进制代码，并继续执行所述第二指令集中所述第四二进制代码之后的二进制代码

图8

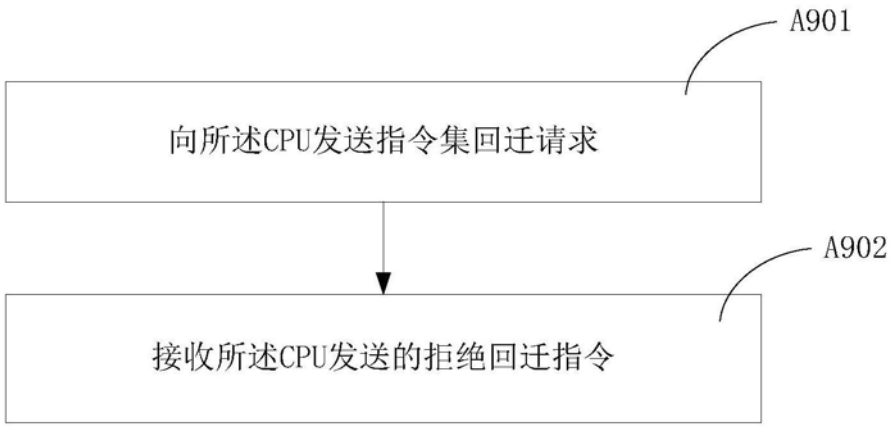

A901
向所述CPU发送指令集回迁请求
A902
接收所述CPU发送的拒绝回迁指令

图9

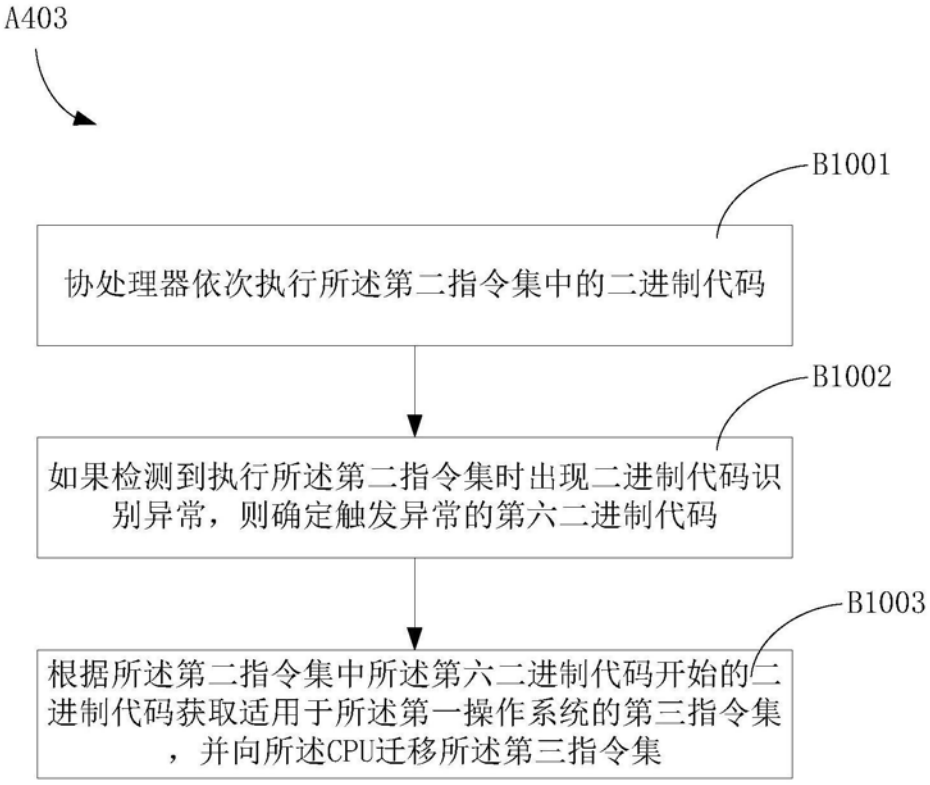

A403
B1001
协处理器依次执行所述第二指令集中的二进制代码
B1002
如果检测到执行所述第二指令集时出现二进制代码识别异常，则确定触发异常的第六二进制代码
B1003
根据所述第二指令集中所述第六二进制代码开始的二进制代码获取适用于所述第一操作系统的第三指令集，并向所述CPU迁移所述第三指令集

图10


B1101
向所述CPU发送指令集回迁请求
B1102
接收所述CPU发送的指令集回迁响应

图11

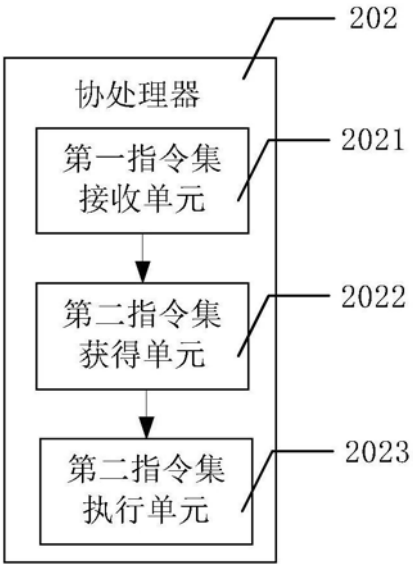

202
协处理器
第一指令集
接收单元
2021
第二指令集
获得单元
2022
第二指令集
执行单元
2023

图12

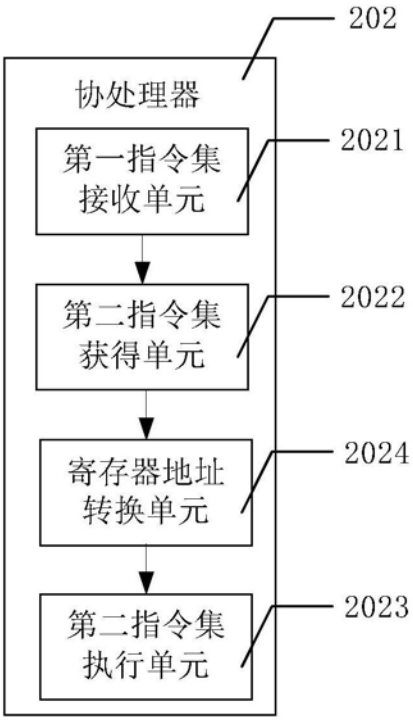

202
协处理器
第一指令集
接收单元
2021
第二指令集
获得单元
2022
寄存器地址
转换单元
2024
第二指令集
执行单元
2023

图13

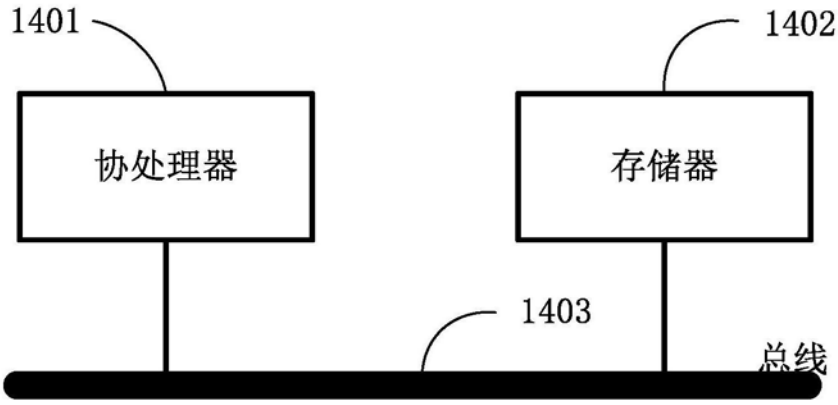



图14