(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 110489213 B
(45)授权公告日 2022.04.05

(21)申请号 201810463722.9
(22)申请日 2018.05.15
(65)同一申请的已公布的文献号
申请公布号 CN 110489213 A
(43)申请公布日 2019.11.22
(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
(72)发明人 李维 高雄 林灏勋 马涛
(74)专利代理机构 北京同达信恒知识产权代理有限公司 11291
代理人 冯艳莲
(51)Int.Cl.
G06F 9/48(2006.01)

(56)对比文件
CN 106874084 A,2017.06.20
CN 101470626 A,2009.07.01
CN 104484167 A,2015.04.01
CN 107665233 A,2018.02.06

审查员 么旭君

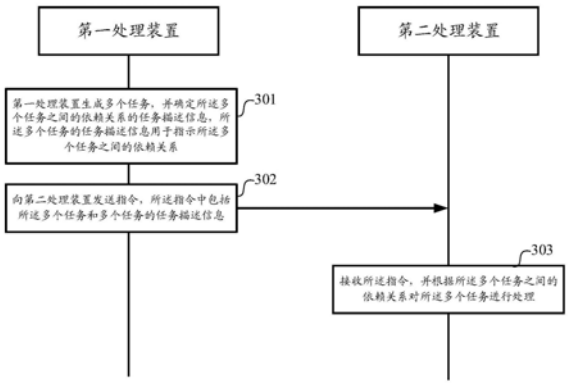
权利要求书4页 说明书15页 附图4页

(54)发明名称

一种任务处理方法及处理装置、计算机系统

(57)摘要

本申请涉及计算机技术领域，公开了一种任务处理方法及处理装置、计算机系统，方法的实现包括：第一处理装置生成多个任务，并确定所述多个任务的任务描述信息，所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系，其中，存在依赖关系的任意两个任务中，其中一个任务的处理需要等待另一个任务的处理结果；所述第一处理装置向第二处理装置发送指令，所述指令中包括所述多个任务和所述多个任务的任务描述信息；所述第二处理装置接收所述指令，并根据所述多个任务之间的依赖关系对所述多个任务进行处理。如此，本申请实施例中的方案能够有效降低等待时延，充分发挥加速芯片的计算能力，提高任务处理效率。



CN 110489213 B

1.一种任务处理方法,其特征在于,所述方法包括:

第一处理装置生成多个任务,并确定所述多个任务的任务描述信息,所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系,其中,存在依赖关系的任意两个任务中,其中一个任务的处理需要等待另一个任务的处理结果;

所述第一处理装置向第二处理装置发送指令,所述指令中包括所述多个任务和所述多个任务的任务描述信息,所述指令用于指示所述第二处理装置根据所述多个任务之间的依赖关系对所述多个任务进行处理。

2.根据权利要求1所述的方法,其特征在于,所述任务描述信息包括所述多个任务分别对应的任务流的标识;

所述第一处理装置通过如下方式确定所述多个任务分别对应的任务流:

所述第一处理装置若确定空闲的任务流的资源量大于等于第一阈值,则从所述空闲的任务流中为所述多个任务分配对应的任务流;或者,

所述第一处理装置若确定空闲的任务流的资源量小于等于第二阈值,则从所述第二处理装置已有的任务流中为所述多个任务分配对应的任务流;或者,

所述第一处理装置若确定空闲的任务流的资源量大于第二阈值且小于第一阈值,则从所述空闲的任务流中为所述多个任务中的部分任务分配对应的任务流,以及从所述第二处理装置已有的任务流中为所述多个任务中的其它部分任务分配对应的任务流。

3.根据权利要求2所述的方法,其特征在于,所述第一处理装置从所述第二处理装置已有的任务流中为所述多个任务分配对应的任务流,包括:

所述第一处理装置根据第一任务需要放入的任务流的优先级,从所述第二处理装置已有的任务流中选择出与所述第一任务需要放入的任务流的优先级相同的任务流,并将选择出的任务流确定为所述第一任务对应的任务流;所述第一任务为所述多个任务中的任一任务。

4.根据权利要求3所述的方法,其特征在于,所述任务描述信息中还包括事件的标识,所述事件用于指示所述事件对应的任务之间的依赖关系,所述事件对应的任务为所述多个任务中位于不同的任务流且具有依赖关系的两个或两个以上的任务,所述两个或两个以上任务包括一个依赖任务以及一个或一个以上被依赖任务;所述依赖任务的处理依赖于所述各个被依赖任务的处理结果。

5.根据权利要求4所述的方法,其特征在于,所述方法还包括:

所述第一处理装置接收所述第二处理装置发送的通知消息,所述通知消息用于指示所述被依赖任务或所述依赖任务处理完成;

所述第一处理装置根据所述通知消息,对所述事件的状态进行更新;

所述第一处理装置若确定所述事件更新后的状态符合预设条件,则释放所述事件的标识。

6.一种任务处理方法,其特征在于,所述方法包括:

第二处理装置接收第一处理装置发送的指令,所述指令中包括多个任务和所述多个任务的任务描述信息,所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系,其中,存在依赖关系的任意两个任务中,其中一个任务的处理需要等待另一个任务的处理结果;

所述第二处理装置根据所述多个任务之间的依赖关系对所述多个任务进行处理。

7.根据权利要求6所述的方法，其特征在于，所述任务描述信息包括事件的标识，所述事件用于指示所述事件对应的任务之间的依赖关系，所述事件对应的任务为所述多个任务中位于不同的任务流且具有依赖关系的两个或两个以上的任务，所述两个或两个以上任务包括一个依赖任务以及一个或一个以上被依赖任务；所述依赖任务的处理依赖于所述各个被依赖任务的处理结果；

所述第二处理装置根据所述多个任务之间的依赖关系对所述多个任务进行处理，包括：

所述第二处理装置确定所述各个被依赖任务均处理完成后，将所述事件的状态更新为就绪状态；

所述第二处理装置确定所述依赖任务为待处理任务后，若检测到所述事件的状态为就绪状态，则对所述依赖任务进行处理。

8.根据权利要求7所述的方法，其特征在于，所述方法还包括：

所述第二处理装置确定所述被依赖任务或所述依赖任务处理完成后，向所述第一处理装置返回通知消息，所述通知消息用于指示所述被依赖任务或所述依赖任务处理完成。

9.一种处理装置，其特征在于，所述处理装置包括：

处理单元，用于生成多个任务，并确定所述多个任务的任务描述信息，所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系，其中，存在依赖关系的任意两个任务中，其中一个任务的处理需要等待另一个任务的处理结果；

收发单元，用于向第二处理装置发送指令，所述指令中包括所述多个任务和所述多个任务的任务描述信息，所述指令用于指示所述第二处理装置根据所述多个任务之间的依赖关系对所述多个任务进行处理。

10.根据权利要求9所述的装置，其特征在于，所述任务描述信息包括所述多个任务分别对应的任务流的标识；

所述处理单元通过如下方式确定所述多个任务分别对应的任务流：

若确定空闲的任务流的资源量大于等于第一阈值，则从所述空闲的任务流中为所述多个任务分配对应的任务流；或者，

若确定空闲的任务流的资源量小于等于第二阈值，则从所述第二处理装置已有的任务流中为所述多个任务分配对应的任务流；或者，

若确定空闲的任务流的资源量大于第二阈值且小于第一阈值，则从所述空闲的任务流中为所述多个任务中的部分任务分配对应的任务流，以及从所述第二处理装置已有的任务流中为所述多个任务中的其它部分任务分配对应的任务流。

11.根据权利要求10所述的装置，其特征在于，所述处理单元具体用于：

根据第一任务需要放入的任务流的优先级，从所述第二处理装置已有的任务流中选择出与所述第一任务需要放入的任务流的优先级相同的任务流，并将选择出的任务流确定为所述第一任务对应的任务流；所述第一任务为所述多个任务中的任一任务。

12.根据权利要求11所述的装置，其特征在于，所述任务描述信息中还包括事件的标识，所述事件用于指示所述事件对应的任务之间的依赖关系，所述事件对应的任务为所述多个任务中位于不同的任务流且具有依赖关系的两个或两个以上的任务，所述两个或两个

以上任务包括一个依赖任务以及一个或一个以上被依赖任务;所述依赖任务的处理依赖于所述各个被依赖任务的处理结果。

13.根据权利要求12所述的装置,其特征在于,所述收发单元还用于:接收所述第二处理装置发送的通知消息,所述通知消息用于指示所述被依赖任务或所述依赖任务处理完成;

所述处理单元还用于:根据所述通知消息,对所述事件的状态进行更新;若确定所述事件更新后的状态符合预设条件,则释放所述事件的标识。

14.一种处理装置,其特征在于,所述处理装置包括:

收发单元,用于接收第一处理装置发送的指令,所述指令中包括多个任务和所述多个任务的任务描述信息,所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系,其中,存在依赖关系的任意两个任务中,其中一个任务的处理需要等待另一个任务的处理结果;

处理单元,用于根据所述多个任务之间的依赖关系对所述多个任务进行处理。

15.根据权利要求14所述的装置,其特征在于,所述任务描述信息包括事件的标识,所述事件用于指示所述事件对应的任务之间的依赖关系,所述事件对应的任务为所述多个任务中位于不同的任务流且具有依赖关系的两个或两个以上的任务,所述两个或两个以上任务包括一个依赖任务以及一个或一个以上被依赖任务;所述依赖任务的处理依赖于所述各个被依赖任务的处理结果;

所述处理单元具体用于:

确定所述各个被依赖任务均处理完成后,将所述事件的状态更新为就绪状态;

确定所述依赖任务为待处理任务后,若检测到所述事件的状态为就绪状态,则对所述依赖任务进行处理。

16.根据权利要求15所述的装置,其特征在于,所述收发单元还用于:

确定所述被依赖任务或所述依赖任务处理完成后,向所述第一处理装置返回通知消息,所述通知消息用于指示所述被依赖任务或所述依赖任务处理完成。

17.一种计算机系统,其特征在于,所述计算机系统包括:第一处理装置和第二处理装置;

所述第一处理装置,用于生成多个任务,并确定所述多个任务的任务描述信息,所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系,其中,存在依赖关系的任意两个任务中,其中一个任务的处理需要等待另一个任务的处理结果;以及,向第二处理装置发送指令,所述指令中包括所述多个任务和所述多个任务的任务描述信息;

所述第二处理装置,用于接收所述指令,并根据所述多个任务之间的依赖关系对所述多个任务进行处理。

18.根据权利要求17所述的计算机系统,其特征在于,所述指令中还包括所述多个任务分别对应的任务流的标识;

所述第一处理装置还用于通过如下方式确定所述多个任务分别对应的任务流:

若确定空闲的任务流的资源量大于等于第一阈值,则从所述空闲的任务流中为所述多个任务分配对应的任务流;或者,

若确定空闲的任务流的资源量小于等于第二阈值,则从所述第二处理装置已有的任务

流中为所述多个任务分配对应的任务流;或者,

若确定空闲的任务流的资源量大于第二阈值且小于第一阈值,则从所述空闲的任务流中为所述多个任务中的部分任务分配对应的任务流,以及从所述第二处理装置已有的任务流中为所述多个任务中的其它部分任务分配对应的任务流。

19.根据权利要求18所述的计算机系统,其特征在于,所述第一处理装置具体用于:

根据第一任务需要放入的任务流的优先级,从所述第二处理装置已有的任务流中选择出与所述第一任务需要放入的任务流的优先级相同的任务流,并将选择出的任务流确定为所述第一任务对应的任务流;所述第一任务为所述多个任务中的任一任务。

20.根据权利要求17至19中任一项所述的计算机系统,其特征在于,所述任务描述信息中还包括事件的标识,所述事件用于指示所述事件对应的任务之间的依赖关系,所述事件对应的任务为所述多个任务中位于不同的任务流且具有依赖关系的两个或两个以上的任务,所述两个或两个以上任务包括一个依赖任务以及一个或一个以上被依赖任务;所述依赖任务的处理依赖于所述各个被依赖任务的处理结果;

所述第二处理装置具体用于:

确定所述各个被依赖任务均处理完成后,将所述事件的状态更新为就绪状态;

确定所述依赖任务为待处理任务后,若检测到所述事件的状态为就绪状态,则对所述依赖任务进行处理。

21.根据权利要求20所述的计算机系统,其特征在于,所述第二处理装置还用于:确定所述被依赖任务或所述依赖任务处理完成后,向所述第一处理装置返回通知消息,所述通知消息用于指示所述被依赖任务或所述依赖任务处理完成;

所述第一处理装置还用于:根据所述通知消息,对所述事件的状态进行更新;以及,若确定所述事件更新后的状态符合预设条件,则释放所述事件的标识。

一种任务处理方法及处理装置、计算机系统

技术领域

[0001] 本申请涉及计算机技术领域，尤其涉及一种任务处理方法及处理装置、计算机系统。

背景技术

[0002] 在信息产业高速发展的今天，计算需求日益丰富，计算机的计算性能不断提升，并由过去的同构计算发展到现在的异构计算。异构计算是指采用了中央处理器(central processing unit,CPU)之外的处理器完成计算任务，不同的计算任务可能采用不同的处理器，比如:数字信号处理器(digital signal processing,DSP)、图形处理器(graphics processing unit,GPU)、现场可编程门阵列(field programmable gate array,FPGA)，以及近年来出现的神经网络处理器(neural network processing unit,NPU)。NPU服务于人工智能(artificial intelligence,AI)计算，特长在于海量的矩阵乘法，可以高效地完成神经网络中的卷积计算。由于异构计算采用多核并行的处理架构，所需的计算时间极低，因此对任务的调度效率提出了更高的要求。

[0003] 当前主流的异构计算均采用主机-设备(Host-Device)模型，由Host向Device异步分发计算任务和相关(输入/输出)数据搬移任务，然后由Device完成计算和数据搬移。其中，Host在向Device分发任务时，需要保证该任务所依赖的前提条件都满足(比如，该任务所依赖的任务均已处理完成)后才分发该任务，而没有相互依赖关系的任务则可以随机发送。具体来说，Host根据应用程序的调用生成多个任务，并把多个任务放入不同的队列中，然后从队列中取任务发送给Device;Host需要在接收到Device返回的处理结果后，再发送下一个任务。

[0004] 然而，由于AI计算任务间的多层数据依赖特征，Device将当前任务处理完成后，需要通过高延迟的Host-Device线再次通知Host获取后继任务，从而增加了任务的等待时延，造成Device的计算停顿，不能充分发挥加速芯片的计算能力。

发明内容

[0005] 本申请提供了一种任务处理方法及处理装置、计算机系统，用以解决任务的等待时延较大、不能充分发挥加速芯片的计算能力的问题。

[0006] 第一方面，本申请实施例提供一种任务处理方法，包括：

[0007] 第一处理装置生成多个任务，并确定所述多个任务的任务描述信息，所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系，其中，存在依赖关系的任意两个任务中，其中一个任务的处理需要等待另一个任务的处理结果；

[0008] 所述第一处理装置向第二处理装置发送指令，所述指令中包括所述多个任务和所述多个任务的任务描述信息。

[0009] 本申请实施例中，第一处理装置将多个任务以及多个任务的任务描述信息一并发送第二处理装置，如此，第二处理装置在处理完成一个任务后，可以直接根据多个任务之间

的依赖关系获取下一个任务进行处理，相比于现有技术中，第二处理装置处理完成一个任务后需要通知给第一处理装置，并由第一处理装置下发下一个任务来说，本申请实施例中的方案能够有效降低等待时延，充分发挥加速芯片的计算能力，提高任务处理效率；且，能够降低第一处理装置的处理负载，满足未来第二处理装置计算能力扩展的场景下，第一处理装置与第二处理装置的负载平衡，并能够更好地满足芯片算力提升、高数据吞吐量、多层数据依赖特征的AI场景下，计算能力与数据搬移能力的平衡和整体性能提升。

[0010] 在一种可能的设计中，所述任务描述信息包括所述多个任务分别对应的任务流的标识；

[0011] 所述第一处理装置通过如下方式确定所述多个任务分别对应的任务流：

[0012] 所述第一处理装置若确定空闲的任务流的资源量大于等于第一阈值，则从所述空闲的任务流中为所述多个任务分配对应的任务流；或者，所述第一处理装置若确定空闲的任务流的资源量小于等于第二阈值，则从所述第二处理装置已有的任务流中为所述多个任务分配对应的任务流；或者，所述第一处理装置若确定空闲的任务流的资源量大于第二阈值且小于第一阈值，则从所述空闲的任务流中为所述多个任务中的部分任务分配对应的任务流，以及从所述第二处理装置已有的任务流中为所述多个任务中的其它部分任务分配对应的任务流。

[0013] 如此，第一处理装置在确定多个任务分别对应的任务流的标识时，会判断第二处理装置中空闲的任务流的标识是否够用，若不够用，则进行共享复用，从而能够有效提高资源利用效率。

[0014] 在一种可能的设计中，所述第一处理装置从所述第二处理装置已有的任务流中为所述多个任务分配对应的任务流，包括：

[0015] 所述第一处理装置根据第一任务需要放入的任务流的优先级，从所述第二处理装置已有的任务流中选择出与所述第一任务需要放入的任务流的优先级相同的任务流，并将选择出的任务流确定为所述第一任务对应的任务流；所述第一任务为所述多个任务中的任一任务。

[0016] 本申请实施例中，由于第二处理装置在对各个任务流中的任务进行处理时，会按照各个任务流的优先级依次进行轮询处理各个任务流的等待列表表头任务。如此，由于上述实现方式充分考虑了各个任务所需要放入的任务流的优先级，从而能够有效保证位于不同任务流中的任务的执行顺序。

[0017] 在一种可能的设计中，所述任务描述信息中还包括事件的标识，所述事件用于指示所述事件对应的任务之间的依赖关系，所述事件对应的任务为所述多个任务中位于不同的任务流且具有依赖关系的两个或两个以上的任务，所述两个或两个以上任务包括一个依赖任务以及一个或一个以上被依赖任务；所述依赖任务的处理依赖于所述各个被依赖任务的处理结果。

[0018] 在一种可能的设计中，所述方法还包括：

[0019] 所述第一处理装置接收所述第二处理装置发送的通知消息，所述通知消息用于指示所述被依赖任务或所述依赖任务处理完成；

[0020] 所述第一处理装置根据所述通知消息，对所述事件的状态进行更新；

[0021] 所述第一处理装置若确定所述事件更新后的状态符合预设条件，则释放所述事件

的标识，以便于后续分配使用。

[0022] 第二方面，本申请实施例提供一种任务处理方法，所述方法包括：

[0023] 第二处理装置接收第一处理装置发送的指令，所述指令中包括多个任务和所述多个任务的任务描述信息，所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系，其中，存在依赖关系的任意两个任务中，其中一个任务的处理需要等待另一个任务的处理结果；

[0024] 所述第二处理装置根据所述多个任务之间的依赖关系对所述多个任务进行处理。

[0025] 由于指令中包括多个任务以及多个任务的任务描述信息，如此，第二处理装置在处理完成一个任务后，可以直接根据多个任务之间的依赖关系获取下一个任务进行处理，能够有效降低等待时延，充分发挥加速芯片的计算能力，提高任务处理效率。

[0026] 在一种可能的设计中，所述任务描述信息包括事件的标识，所述事件用于指示所述事件对应的任务之间的依赖关系，所述事件对应的任务为所述多个任务中位于不同的任务流且具有依赖关系的两个或两个以上的任务，所述两个或两个以上任务包括一个依赖任务以及一个或一个以上被依赖任务；所述依赖任务的处理依赖于所述各个被依赖任务的处理结果；

[0027] 所述第二处理装置根据所述多个任务之间的依赖关系对所述多个任务进行处理，包括：

[0028] 所述第二处理装置确定所述各个被依赖任务均处理完成后，将所述事件的状态更新为就绪状态；

[0029] 所述第二处理装置确定所述依赖任务为待处理任务后，若检测到所述事件的状态为就绪状态，则对所述依赖任务进行处理。

[0030] 在一种可能的设计中，所述方法还包括：

[0031] 所述第二处理装置确定所述被依赖任务或所述依赖任务处理完成后，向所述第一处理装置返回通知消息，所述通知消息用于指示所述被依赖任务或所述依赖任务处理完成。

[0032] 第三方面，本申请实施提供一种处理装置，该装置具有实现上述第一方面方法示例中第一处理装置所执行的行为的功能。所述功能可以通过硬件实现，也可以通过硬件执行相应的软件实现。所述硬件或所述软件包括一个或多个与上述功能相对应的单元(或模块)。

[0033] 在一个可能的设计中，所述装置的结构中包括：处理单元和收发单元；其中，处理单元，用于生成多个任务，并确定所述多个任务的任务描述信息，所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系，其中，存在依赖关系的任意两个任务中，其中一个任务的处理需要等待另一个任务的处理结果；

[0034] 收发单元，用于向第二处理装置发送指令，所述指令中包括所述多个任务和所述多个任务的任务描述信息。

[0035] 在一种可能的设计中，所述任务描述信息包括所述多个任务分别对应的任务流的标识；

[0036] 所述处理单元通过如下方式确定所述多个任务分别对应的任务流：

[0037] 若确定空闲的任务流的资源量大于等于第一阈值，则从所述空闲的任务流中为所

述多个任务分配对应的任务流;或者,若确定空闲的任务流的资源量小于等于第二阈值,则从所述第二处理装置已有的任务流中为所述多个任务分配对应的任务流;或者,若确定空闲的任务流的资源量大于第二阈值且小于第一阈值,则从所述空闲的任务流中为所述多个任务中的部分任务分配对应的任务流,以及从所述第二处理装置已有的任务流中为所述多个任务中的其它部分任务分配对应的任务流。

[0038] 在一种可能的设计中,所述处理单元具体用于:

[0039] 根据第一任务需要放入的任务流的优先级,从所述第二处理装置已有的任务流中选择出与所述第一任务需要放入的任务流的优先级相同的任务流,并将选择出的任务流确定为所述第一任务对应的任务流;所述第一任务为所述多个任务中的任一任务。

[0040] 在一种可能的设计中,所述任务描述信息中还包括事件的标识,所述事件用于指示所述事件对应的任务之间的依赖关系,所述事件对应的任务为所述多个任务中位于不同的任务流且具有依赖关系的两个或两个以上的任务,所述两个或两个以上任务包括一个依赖任务以及一个或一个以上被依赖任务;所述依赖任务的处理依赖于所述各个被依赖任务的处理结果。

[0041] 在一种可能的设计中,所述收发单元还用于:接收所述第二处理装置发送的通知消息,所述通知消息用于指示所述被依赖任务或所述依赖任务处理完成;

[0042] 所述处理单元还用于:根据所述通知消息,对所述事件的状态进行更新;若确定所述事件更新后的状态符合预设条件,则释放所述事件的标识。

[0043] 第四方面,本申请实施提供一种处理装置,该装置具有实现上述第二方面方法示例中第二处理装置所执行的行为的功能。所述功能可以通过硬件实现,也可以通过硬件执行相应的软件实现。所述硬件或所述软件包括一个或多个与上述功能相对应的单元(或模块)。

[0044] 在一个可能的设计中,所述装置的结构中包括:处理单元和收发单元;其中,收发单元,用于接收第一处理装置发送的指令,所述指令中包括多个任务和所述多个任务的任务描述信息,所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系,其中,存在依赖关系的任意两个任务中,其中一个任务的处理需要等待另一个任务的处理结果;

[0045] 处理单元,用于根据所述多个任务之间的依赖关系对所述多个任务进行处理。

[0046] 在一种可能的设计中,所述任务描述信息包括事件的标识,所述事件用于指示所述事件对应的任务之间的依赖关系,所述事件对应的任务为所述多个任务中位于不同的任务流且具有依赖关系的两个或两个以上的任务,所述两个或两个以上任务包括一个依赖任务以及一个或一个以上被依赖任务;所述依赖任务的处理依赖于所述各个被依赖任务的处理结果;

[0047] 所述处理单元具体用于:

[0048] 确定所述各个被依赖任务均处理完成后,将所述事件的状态更新为就绪状态;

[0049] 确定所述依赖任务为待处理任务后,若检测到所述事件的状态为就绪状态,则对所述依赖任务进行处理。

[0050] 在一种可能的设计中,所述收发单元还用于:

[0051] 确定所述被依赖任务或所述依赖任务处理完成后,向所述第一处理装置返回通知消息,所述通知消息用于指示所述被依赖任务或所述依赖任务处理完成。

[0052] 第五方面,本申请实施例提供一种处理装置,所述处理装置包括:

[0053] 存储器,用于存储软件程序;

[0054] 处理器,用于读取所述存储器中的软件程序,并执行上述第一方面或第二方面的任意一种可能的设计中所述的方法。

[0055] 本申请实施例还提供一种计算机系统,所述计算机系统包括:第一处理装置和第二处理装置;第一处理装置具有实现上述第一方面方法示例中第一处理装置所执行的行为的功能,第二处理装置具有实现上述第二方面方法示例中第二处理装置所执行的行为的功能,此处不再赘述。

[0056] 本申请实施例还提供一种计算机存储介质,该存储介质中存储软件程序,该软件程序在被第一处理装置读取并执行时,可实现上述第一方面的任意一种可能的设计提供的方法。

[0057] 本申请实施例还提供一种计算机存储介质,该存储介质中存储软件程序,该软件程序在被第二处理装置读取并执行时可实现上述第二方面的任意一种可能的设计提供的方法。

[0058] 本申请实施例还提供一种包含指令的计算机程序产品,当其在计算机上运行时,使得计算机执行上述第一方面或第二方面的任意一种可能的设计提供的方法。

附图说明

[0059] 图1为现有技术中采用Host-Device模型进行任务处理的示例图;

[0060] 图2为本申请实施例提供的一种计算机系统架构示意图;

[0061] 图3为本申请实施例提供的一种任务处理方法所对应的流程示意图;

[0062] 图4为本申请实施例提供的Host与Device的调度配合示意图;

[0063] 图5为本申请实施例应用的计算机的硬件结构示意图;

[0064] 图6为本申请实施提供的一种处理装置的结构示意图;

[0065] 图7为本申请实施提供的又一种处理装置的结构示意图。

具体实施方式

[0066] 首先,对本申请中的部分用语进行解释说明,以便使本领域技术人员理解。

[0067] 计算机系统:集成至少两个异构装置的系统,例如可以为一个集成至少两个异构装置的芯片,或一个集成至少两个异构装置的异构集成平台。

[0068] 异构装置:不同类型的处理装置,举例而言,CPU、GPU、FPGA以及ASIC等不同类型的处理装置之间可互称为异构装置。在本发明的其他实施例中,同一类型的处理设备在具有不同规格时,也可互称为异构装置,例如具有不同主频的CPU也可互称为异构装置。

[0069] 任务(task):是指由软件完成的一个活动。一个任务既可以是一个进程,也可以是一个线程。简而言之,它指的是一系列共同达到某一目的的操作。本申请实施例中的任务可以包括计算任务、内存拷贝(memcpy)任务、虚拟任务等,其中,虚拟任务是指事件发生(EventRecord)(如某个任务的完成)或事件等待(EventWait)(如需要等待某个任务完成才能继续)。

[0070] 流(stream):包含多个任务的任务流,其内部的任务必须按照顺序执行。举个例

子,流中先后挂入3个任务,分别为任务1、任务2、任务3,则在执行时,需按照任务1、任务2、任务3的顺序进行执行,即先执行任务1,在任务1执行完成后,开始执行任务2,在任务2执行完成后,开始执行任务3。

[0071] 多个:指两个或两个以上。

[0072] 为了使本申请的目的、技术方案和优点更加清楚,下面将结合附图对本申请作进一步地详细描述。

[0073] 图1为现有技术中采用Host-Device模型进行任务处理的示例图。如图1所示,Host生成多个任务(比如,图1中所示意出的任务1、任务2、…、任务9),并根据9个任务之间的依赖关系,将9个任务分别放入对应的队列中,比如,将任务1、任务2和任务3放入队列1中,将任务4、任务5和任务6放入队列2中,将任务7、任务8和任务9放入队列3中。其中,任务之间带有箭头的线段用于表示任务之间的依赖关系,线段的箭头所指向的任务依赖于线段另一端的任务,比如任务2依赖于任务3,任务5依赖于任务3,即任务3的处理顺序先于任务2,且先于任务5。

[0074] Host侧部署有任务级调度器,任务级调度器从各个队列中取任务下发给Device侧,比如,任务级调度器从队列1中调度任务3下发给Device侧;Device侧接收到任务3(假设任务3为计算任务)后,可以由线程块级调度器将任务3拆分为多个线程块(Block),并分发到多个计算核(Core)上并行计算。当每一个计算核都完成各自的计算任务后,线程块级调度器向Host发送消息,通知任务3完成。随后,Host侧可调度下一个任务(任务2)下发给Device侧。

[0075] 根据上述过程可知,由于Host需要在接收到Device返回的处理结果后,再发送下一个任务,从而导致任务的等待时延较大,不能充分发挥加速芯片的计算能力。

[0076] 基于此,本申请实施例提供一种任务处理方法,用于解决计算任务的等待时延较大、不能充分发挥加速芯片的计算能力的问题。

[0077] 本申请实施例提供的任务处理方法,可以适用于如图2所示的计算机系统架构中,该计算机系统架构包括第一处理装置210、第二处理装置220。

[0078] 第一处理装置210可以为Host,第一处理装置210中部署有设备运行时(Device Runtime)和设备驱动(Device Driver);其中,设备运行时可用于向第二处理装置下发任务。

[0079] 第二处理装置220可以为Device,第二处理装置220中部署有用于计算的加速库(Acc.Core),还部署了一个微控制单元(microcontroller unit,MCU),如:ARM core。微控制单元上运行轻量的操作系统(Lite OS),该轻量的操作系统上运行第二处理装置220本地的调度软件,这里称作“Taskscheduler”。其中,Taskscheduler负责对从第一处理装置接收到的任务进行调度和分发。

[0080] 处理器(核)间通信(inter-processor communication,IPC)是第一处理装置与第二处理装置之间的物理连接,具体形态不限,比如:PCI-E。

[0081] 基于图2所示意的系统架构,图3为本申请实施例提供的一种任务处理方法所对应的流程示意图,如图3所示,该方法包括:

[0082] 步骤301,第一处理装置生成多个任务,并确定所述多个任务的任务描述信息,所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系,其中,存在依赖关系

的任意两个任务中，其中一个任务的执行需要等待另一个任务的执行结果；

[0083] 步骤302，第一处理装置向第二处理装置发送指令，所述指令中包括所述多个任务和所述多个任务的任务描述信息；

[0084] 步骤303，第二处理装置接收所述指令，并根据所述多个任务之间的依赖关系对所述多个任务进行处理。

[0085] 根据上述内容可知，本申请实施例中第一处理装置将多个任务以及多个任务的任务描述信息一并发送第二处理装置，如此，第二处理装置在处理完成一个任务后，可以直接根据多个任务之间的依赖关系获取下一个任务进行处理，相比于现有技术中，第二处理装置处理完成一个任务后需要通知给第一处理装置，并由第一处理装置下发下一个任务来说，本申请实施例中的方案能够有效降低等待时延，充分发挥加速芯片的计算能力，提高任务处理效率；且，能够降低第一处理装置的处理负载，满足未来第二处理装置计算能力扩展的场景下，第一处理装置与第二处理装置的负载平衡，并能够更好地满足芯片算力提升、高数据吞吐量、多层数据依赖特征的AI场景下，计算能力与数据搬移能力的平衡和整体性能提升。

[0086] 具体来说，第一处理装置可以为CPU，CPU的软件栈从底层至上层包括：操作系统、驱动或运行时(runtime)、程序框架和业务代码。其中，业务代码通俗而言即为应用程序，例如语音识别程序、图像处理程序等。应用程序通过调用操作系统的应用程序编程接口(application programming interface，API)而使操作系统去执行应用程序的命令(动作)。本申请实施例的步骤301中，第一处理装置生成多个任务具体可以为第一处理装置中的应用程序通过调用操作系统的API而使操作系统生成多个任务。需要说明的是，第一处理装置中的应用程序可以通过调用操作系统的一个API而使操作系统生成多个任务，或者，也可以是通过调用操作系统的多个API而使操作系统生成多个任务。进一步地，第一处理装置还可以基于开发者的描述确定出多个任务之间的依赖关系。

[0087] 举个例子，第一处理装置生成的多个任务为任务1、任务2和任务3，其中，任务1是由第一处理装置中的应用程序通过调用操作系统的第一API而生成的，任务2和任务3是由第一处理装置中的应用程序通过调用操作系统的第二API而生成的。多个任务之间的依赖关系为：任务3依赖于任务1和任务2，也就是说，任务1和任务2的处理顺序先于任务3，而由于任务1和任务2是通过调用不同的API生成的，因此，任务1和任务2可以并发处理(假设不存在其它约束条件)。

[0088] 第二处理装置可以为GPU，步骤302中，第一处理装置发送给第二处理装置的任务描述信息中可以包括所述多个任务分别对应的任务流的标识。

[0089] 具体来说，所述多个任务分别对应的任务流可以是通过如下方式确定的：第一处理装置若确定空闲的任务流的资源量大于等于第一阈值，则从所述空闲的任务流中为所述多个任务分配对应的任务流；或者，第一处理装置若确定空闲的任务流的资源量小于等于第二阈值，则从所述第二处理装置已有的任务流为所述多个任务分配对应的任务流；或者，第一处理装置若确定空闲的任务流的资源量大于第二阈值且小于第一阈值，则从所述空闲的任务流中为所述多个任务中的部分任务分配对应的任务流，以及从所述第二处理装置已有的任务流中为所述多个任务中的其它部分任务分配对应的任务流。

[0090] 也就是说，第一处理装置在确定多个任务分别对应的任务流的标识时，需要判断

空闲的任务流的资源量是否够用。在一种可能的实现方式中，第一处理装置中可以维护有任务流列表，如表1所示。

[0091] 表1：任务流列表示例

[0092]

任务流的标识	任务流的状态（streamstate）
stream 1	……
stream 2	……
……	……
stream m	……

[0093] 具体来说，第一处理装置可以根据第二处理装置中各个任务流对应的内存，更新各个任务流的状态(streamstate)，任务流的状态具体可以为该任务流中驻留的任务的个数。

[0094] 本申请实施例中，任务流的个数可以为预先设定的，第一处理装置可以表1中任务流的个数和预先设定的任务流的个数来判断空闲的任务流的资源量是否够用。其中，空闲的任务流的资源量具体可以为空闲的任务流的标识的个数，任务流的标识可以为用于唯一标识任务流的信息，比如任务流的流水号。在一个示例中，设预先设定任务流的个数为100个，若表1中任务流的个数为100个，则说明此时暂无空闲的任务流的标识；若表1中任务流的个数为99个，则说明此时存在1个空闲的任务流的标识。

[0095] 比如，上述所示例的任务1、任务2、任务3，由于任务1和任务2、任务3是通过不同的API调用的，因此需要为任务1、任务2、任务3需要分别放入两个任务流(任务1放入一个任务流，任务2和任务3放入另一个任务流)。

[0096] 第一处理装置若确定空闲的任务流的标识的个数大于等于第一阈值(第一阈值可以根据实际需要进行设置，在此示例中，第一阈值可以为大于等于2的数值，比如2)，则说明第二处理装置中空闲的任务流的标识完全够用，此时，第一处理装置可将任意两个空闲的任务流的标识分别分配给任务1和任务2、任务3，比如，将空闲的任务流的标识(stream 1)分配给任务1，将空闲的任务流的标识(stream 2)分配给任务2和任务3。如此，第二处理装置接收到第一处理装置发送的指令后，可根据指令中所包括的任务1对应的任务流的标识(stream 1)，将任务1放入标识为stream 1的任务流中；以及，根据指令中所包括的任务2和任务3对应的任务流的标识(stream 2)，将任务2和任务3放入标识为stream 2的任务流中。

[0097] 第一处理装置若确定空闲的任务流的标识的个数小于等于第二阈值(第二阈值可以根据实际需要进行设置，在此示例中，第二阈值可以为0)，则说明此时暂无空闲的任务流的标识，因此第一处理装置可从所述第二处理装置中已有的任务流的标识中为所述多个任务分配对应的任务流的标识，即共享复用已有的任务流，比如，将第二处理装置中已有的任务流1(stream 1)确定为任务1对应的任务流，将已有的任务流2(stream 2)确定为任务2和任务3对应的任务流。如此，第二处理装置接收到第一处理装置发送的指令后，可根据指令中所包括的任务1对应的任务流的标识(stream 1)，将任务1放入已有的任务流1(stream 1)中，将任务2和任务3放入已有的任务流2(stream 2)中。

[0098] 本申请实施例中，第一处理装置可以根据第二处理装置中各个已有的任务流的优

先级，来确定多个任务对应的任务流。

[0099] 比如，针对于任务1来说，第一处理装置可以根据任务1所需要放入的任务流的优先级(比如优先级为1)，从各个已有的任务流中选择出优先级为1的任务流(比如任务流1)，若此时任务流1中可以继续容纳任务，则可任务流1确定为任务1对应的任务流中，相应地，第二处理装置在接收到指令后，可将任务1放入已有的任务流1中；若此时任务流1中无法继续容纳任务，则无法进行共享复用，可阻塞任务1，直至有空闲的任务流的标识，或者任务流1中可继续容纳任务。

[0100] 针对于任务2和任务3来说，第一处理装置可以根据任务2和任务3所需要放入的任务流的优先级(比如优先级为2)，从各个已有的任务流中选择出优先级为2的任务流(比如任务流2)，若此时任务流2可继续两个容纳任务，则可将任务流2确定为任务2和任务3对应的任务流，相应地，第二处理装置在接收到指令后，可将任务2和任务3放入已有的任务流2中；若此时任务流2仅可继续容纳一个任务，则将任务流2确定为任务2对应的任务流，并阻塞任务3，直至任务流2中可继续容纳任务，相应地，第二处理装置在接收到指令后，可将任务2放入已有的任务流2中；若此时任务流2无法继续容纳任务，则可阻塞任务2和任务3，直至有空闲的任务流的标识，或者任务流2中可继续容纳任务。

[0101] 本申请实施例中，第二处理装置在对各个任务流中的任务进行处理时，会按照各个任务流的优先级依次进行轮询处理各个任务流的等待列表表头任务。如此，由于上述实现方式充分考虑了各个任务所需要放入的任务流的优先级，从而能够有效保证位于不同任务流中的任务的执行顺序。且，第一处理装置生成任务1、任务2和任务3(任务1所需要放入的任务流的优先级为1，任务2和任务3所需要放入的任务流的优先级为2)，若确定第二处理装置任务流的标识的个数小于等于第二阈值，且第二处理装置中已有的任务流中优先级为1的任务流无法继续容纳任务(即任务1无法共享复用已有的任务流)，优先级为2的任务流也无法继续容纳任务(即任务2和任务3无法共享复用已有的任务流)，则可阻塞所述多个任务(任务1、任务2和任务3)的分发，即将多个任务反压在第一处理装置中，从而减少对第二处理装置内存资源占用，更好地适应第二处理装置内存资源受限的场景，进一步地，还可以方便第一处理装置进行全局异构资源平衡调度。

[0102] 第一处理装置若确定所述第二处理装置中任务流的标识的个数大于第二阈值且小于第一阈值，则说明第二处理装置中空闲的任务流的标识不够用(设空闲的任务流的标识仅有1个)，因此第一处理装置可以将空闲的任务流的标识(stream 1)分配给任务1，以及将第二处理装置中已有的任务流(比如任务流2，任务流2的标识为stream 2)确定为任务2和任务3对应的任务流。如此，第二处理装置接收到第一处理装置发送的指令后，可根据指令中所包括的任务1对应的任务流的标识(stream 1)，将任务1放入标识为任务流1中，以及，根据指令中所包括的任务2和任务3对应的任务流的标识(stream 2)，将任务2和任务3放入已有的任务流2中。

[0103] 此处，需要说明的是，第二处理装置中可能存在多个已有的任务流，第一处理装置从多个已有的任务流中选择任务2和任务3需要共享复用的任务流时，可以根据任务2和任务3所需要放入的任务流的优先级进行选择，具体可参见上述描述。

[0104] 本申请实施例中，第一处理装置也可以将空闲的任务流的标识分配给任务2和任务3，并从已有的任务流选择出任务1对应的任务流。具体实施中，可以依据实际情况进行合

理灵活调整,具体不做限定。

[0105] 通过上述内容可知,第一处理装置在确定多个任务分别对应的任务流的标识时,会判断第二处理装置中空闲的任务流的标识是否够用,若不够用,则进行共享复用,从而能够有效提高资源利用效率。

[0106] 在其它可能的实现方式中,第一处理装置若确定第二处理装置中空闲的任务流的标识不够用,也可以不进行共享复用,直接阻塞直至有空闲的任务流的标识,或者,也可以直接错误返回。具体不做限定。

[0107] 进一步地,根据上述所示例的任务1、任务2和任务3的依赖关系可知,任务3除依赖于任务2之外,还依赖于任务1。由于任务2和任务3位于同一任务流中,因此,可按先后顺序将任务2和任务3放入任务流中,从而能够保证先处理任务2再处理任务3。针对于位于不同任务流中的任务1和任务3,本申请实施例中可以通过事件(event)来指示依赖关系。下面对具体实现过程进行说明。

[0108] 开发者可以根据生成的任务之间的依赖关系来创建事件,具体实施中,开发者通过编写业务代码,进而实现由第一处理装置中的应用程序调用操作系统的API生成事件。所述事件用于指示所述事件对应的任务之间的依赖关系,所述事件对应的任务为所述多个任务中位于不同的任务流且具有依赖关系的两个或两个以上的任务,所述两个或两个以上任务包括一个依赖任务以及一个或一个以上被依赖任务;所述依赖任务的处理依赖于所述各个被依赖任务的处理结果。比如,上述示例中,开发者可以根据任务1和任务3之间的依赖关系创建事件,此时,所述事件对应的任务即为任务1和任务3,任务3为依赖任务,任务1为被依赖任务。

[0109] 本申请实施例中,第一处理装置和第二处理装置中可以同步维护一个事件列表,如表2所示,为事件列表示例。

[0110] 表2:事件列表示例

[0111]

事件的标识	事件的状态 (eventstate)
event 1	……
event 2	……
……	……
event n	……

[0112] 本申请实施例中,事件的标识可以为能够唯一标识一个事件的信息,比如事件的流水号,具体不做限定。

[0113] 基于上述示例,第一处理装置生成任务1、任务2和任务3后,若确定存在空闲的事件的标识,则可生成任务1和任务3对应的事件的标识(比如event n+1),并将该事件的标识添加到第一处理装置所维护的事件列表中,以及将该事件的标识通过指令发送给第二处理装置。

[0114] 相应地,第二处理装置接收到指令后,可根据将指令中包括的事件的标识(event n+1)添加到第二处理装置所维护的事件列表中,此时,该事件对应的状态可以为一个默认状态,具体不做限定。进一步地,第二处理装置确定该事件对应的任务中的被依赖任务(任

务1)处理完成后,将该事件对应的状态更新为就绪状态;后续第二处理装置确定该事件对应的任务中的依赖任务(任务3)为待处理任务后,若检测到该事件的状态为就绪状态,则对所述依赖任务进行处理;否则,等待,直到检测到该事件的状态为就绪状态。

[0115] 本申请实施例中,第二处理装置确定任务处理完成后,还可以向第一处理装置返回通知消息,所述通知消息用于指示所述任务处理完成,从而便于第一处理装置根据第二处理装置返回的通知消息对事件的状态进行更新,进而对事件的标识进行管理。比如,第二处理装置向第一处理装置返回通知消息,该通知消息用于指示被依赖任务处理完成,则第一处理装置根据所述通知消息,可对所述事件的状态进行更新;又比如,第二处理装置向第一处理装置返回通知消息,该通知消息用于指示依赖任务处理完成,则第一处理装置根据所述通知消息,可对所述事件的状态再次进行更新。进一步地,第一处理装置在对事件的状态进行更新后,若确定所述事件更新后的状态符合预设条件后,则释放所述事件的标识,以便于后续分配使用。其中,预设条件可以根据实际需要进行设置。

[0116] 进一步地,第一处理装置确定释放所述事件的标识后,还可以向第二处理装置发送同步消息,该同步消息用于指示第二处理装置释放所述事件的标识,从而保证第一处理装置中维护的事件列表和第二处理装置中维护的事件列表的一致性。

[0117] 在一种可能的实现方式中,第一处理装置可以以Lazy分配方式对事件的标识进行管理,从而提升事件的标识的周转效率。本申请实施例中,可以将事件发生(EventRecord)和事件等待(EventWait)抽象为虚拟任务,等同于计算任务和内存拷贝任务,并为每个虚拟任务分配一个进程内唯一的任务标识,以便于实现下发与上报的任务对应。具体来说,第一处理装置可以引用计数对事件的标识进行管理,当计数为0时,释放事件的标识。一种可能的计数方式如下:

[0118] (1)事件等待:第一处理装置发送指令后,计数加1;第二处理装置执行事件等待后,向第一处理装置返回执行结果,此时,第一处理装置在接收到执行结果后,计数减1。

[0119] (2)事件发生:第一处理装置发送指令后,计数加1;第二处理装置执行事件发生后,向第一处理装置返回执行结果,此时,第一处理装置在接收到执行结果后,计数减1。

[0120] 在上述示例中,第一处理装置向第二处理装置发送的指令中还可以包括虚拟任务(事件发生,即任务1的完成;事件等待,即任务3需要等待)。由于指令中包括事件发生和事件等待,因此,第一处理装置发送指令后,事件的计数变为2(事件等待加1,事件发生加1),后续接收到事件发生的执行结果后,计数减1,接收到事件等待的执行结果后,计数再次减1,并变为0,此时,可认为该事件的状态符合预设条件,第一处理装置可释放事件的标识。

[0121] 需要说明的是,第一处理装置若确定不存在空闲的事件的标识,则可以阻塞并等待,直至有空闲的事件的标识,或者,也可以错误返回。本申请实施例中,第一处理装置确定是否存在空闲的事件的标识的方式可以有多种,比如,第一处理装置可以根据预先设定事件的个数以及事件列表中包括的事件的个数来判断,若预先设定事件的个数为1024个,事件列表中的事件个数也为1024个,则说明不存在空闲的事件的标识;若预先设定事件的个数为1024个,事件列表中的事件个数也为1000个,则说明存在空闲的事件的标识。

[0122] 本申请实施例中,若第一处理装置中进程出现异常退出,则可以由Host操作系统或者内核态驱动(driver)清理任务流和事件资源。

[0123] 基于上文的介绍,图4为本申请实施例提供的Host与Device的调度配合示意图。下

面结合图4对本申请实施例的整体实现过程进行描述。

[0124] 在Host上,运行时(Runtime)提供API,供开发者描述并发的任务。stream、event是任务赖以运行的上下文环境。运行时会保存每个已创建的stream和event事件的状态。运行时在收到计算任务的API调用时,立即将任务发到Device一侧,并附带上各个任务之间的依赖关系(或者,也可以理解为附带上各个任务的执行条件,如某个事件发生)。虽然Host将任务推送到Device一侧,但并不意味着任务可以立即执行。Host上的处理可以理解为对并发多任务流中的任务的一个编排,即对任务进行前端调度。

[0125] 在Device上,会维护多个任务流,TaskScheduler会检测每个任务流的队列头获取待命的任务,即对任务进行后端调度。具体来说,TaskScheduler可以根据各个任务流的优先级(优先级可以为预先设置,此处不做具体限定),依次轮询处理每个任务流的等待列表表头任务,如果该任务为计算任务,且内核调度时隙(Slot)有空闲,则通过线程块级调度器将该任务发送给计算核进行调度执行,具体执行方式可参见上述图1中的描述,此处不再赘述;如果为内存拷贝任务,且有空闲直接内存访问(direct memory access,DMA)通道,则发送给DMA进行执行;如果为事件等待任务,则检查对应事件的状态是否为就绪状态;如果为事件发生任务,则设置对应的事件的状态为就绪状态。进一步地,为了维护事件资源,本申请实施例中还可以增加事件销毁(EventDestroy)任务(与事件创建任务相对应):如果为事件销毁任务,则设置对应的事件的状态为非就绪状态。Taskscheduler同时也会记录事件是否已经发生的状态。这样,当一个先决条件满足后(比如某个任务完成),TaskScheduler可以在本地快速地调度下一个任务,而不需要跨越IPC通知Host。

[0126] 在一个可能的示例中,TaskScheduler所执行的功能可以通过软件来实现,线程块级调度器所执行的功能可以通过专用集成电路(application specific integrated circuit,ASIC)来实现。

[0127] 根据上述内容可知,本申请实施例将任务调度分为Host侧的前端调度和Device侧的后端调度两部分,前端调度负责任务之间的依赖关系的描述、下发,后端调度负责任务的调度执行,从而能够降低Host的处理负载,满足未来Device计算能力扩展场景下,Host与Device负载平衡。进一步地,在Device侧引入CPU处理核和DMA硬件,前者部署后端调度功能,后者执行数据搬移任务,从而提高任务处理效率。

[0128] 针对上述方法流程,本申请实施例还提供一种计算机系统,该计算机系统包括第一处理装置和第二处理装置,第一处理装置和第二处理装置可以实现上述图3示例的方法流程中相应功能,具体参见方法示例中的详细描述,此处不做赘述。

[0129] 需要说明的是,本申请中的第一处理器装置和第二处理装置可以是两个独立的芯片,比如第一处理装置是CPU,第二处理器装置是GPU,第一处理器装置和第二处理装置也可以是两个不同的电路模块,两者集成在一起形成一个芯片,比如,第二处理装置是一个NPU,其和第一处理器CPU集成在一起形成一个芯片。

[0130] 图5为本申请实施例应用的计算机的硬件结构示意图。如图5所示,计算机500包括显示设备510、处理器520以及存储器530。

[0131] 存储器530可用于存储软件程序以及数据,处理器520通过运行存储在存储器530的软件程序以及数据,从而执行计算机500的各种功能应用以及数据处理。存储器530可主要包括存储程序区和存储数据区,其中,存储程序区可存储操作系统信息、至少一个功能所

需的应用程序(比如数值计算功能等)等;存储数据区可存储根据计算机500的使用所创建的数据(比如音频数据、图像数据等)等。此外,存储器530可以包括高速随机存取存储器,还可以包括非易失性存储器,例如至少一个磁盘存储器件、闪存器件、或其他易失性固态存储器件。处理器520是计算机500的控制中心,利用各种接口和线路连接整个计算机的各个部分,通过运行或执行存储在存储器130内的软件程序和/或数据,执行计算机500的各种功能和处理数据,从而对计算机进行整体监控。处理器520可以包括一个或多个通用处理器,还可包括一个或多个GPU,用于执行相关操作,以实现本申请实施例所提供的技术方案。

[0132] 计算机500中还包括用于拍摄图像或视频的摄像头560。摄像头560可以是普通摄像头,也可以是对焦摄像头。

[0133] 计算机500还可以包括输入设备540,用于接收输入的数字信息、字符信息或接触式触摸操作/非接触式手势,以及产生与计算机500的用户设置以及功能控制有关的信号输入等。

[0134] 显示设备510,包括的显示面板511,用于显示由用户输入的信息或提供给用户的信息以及计算机500的各种菜单界面等,可选的,显示面板可以采用液晶显示器(liquid crystal display,LCD)或OLED(organic light-emitting diode,有机发光二极管)等形式来配置显示面板511。

[0135] 除以上之外,计算机500还可以包括用于给其他模块供电的电源550。计算机500还可以包括一个或多个传感器570,例如图像传感器、红外传感器、激光传感器等。计算机500还可以包括无线射频(radio frequency,RF)电路580,用于与无线网络设备进行网络通信,还可以包括WiFi模块590,用于与其他设备进行WiFi通信,获取其他设备传输的图像或者数据等。

[0136] 图6为本申请实施提供的一种处理装置的结构示意图,该装置具有实现上述图3所示意的方法流程中第一处理装置所执行的行为的功能。所述处理装置600中包括:处理单元601和收发单元602;其中,处理单元601,用于生成多个任务,并确定所述多个任务的任务描述信息,所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系,其中,存在依赖关系的任意两个任务中,其中一个任务的处理需要等待另一个任务的处理结果;

[0137] 收发单元602,用于向第二处理装置发送指令,所述指令中包括所述多个任务和所述多个任务的任务描述信息。

[0138] 在一种可能的设计中,所述任务描述信息包括所述多个任务分别对应的任务流的标识;

[0139] 所述处理单元601通过如下方式确定所述多个任务分别对应的任务流:

[0140] 若确定空闲的任务流的资源量大于等于第一阈值,则从所述空闲的任务流中为所述多个任务分配对应的任务流;或者,

[0141] 若确定空闲的任务流的资源量小于等于第二阈值,则从所述第二处理装置已有的任务流中为所述多个任务分配对应的任务流;或者,

[0142] 若确定空闲的任务流的资源量大于第二阈值且小于第一阈值,则从所述空闲的任务流中为所述多个任务中的部分任务分配对应的任务流,以及从所述第二处理装置已有的任务流中为所述多个任务中的其它部分任务分配对应的任务流。

[0143] 在一种可能的设计中,所述处理单元601具体用于:

[0144] 根据第一任务需要放入的任务流的优先级，从所述第二处理装置已有的任务流中选择出与所述第一任务需要放入的任务流的优先级相同的任务流，并将选择出的任务流确定为所述第一任务对应的任务流；所述第一任务为所述多个任务中的任一任务。

[0145] 在一种可能的设计中，所述任务描述信息中还包括事件的标识，所述事件用于指示所述事件对应的任务之间的依赖关系，所述事件对应的任务为所述多个任务中位于不同的任务流且具有依赖关系的两个或两个以上的任务，所述两个或两个以上任务包括一个依赖任务以及一个或一个以上被依赖任务；所述依赖任务的处理依赖于所述各个被依赖任务的处理结果。

[0146] 在一种可能的设计中，所述收发单元602还用于：接收所述第二处理装置发送的通知消息，所述通知消息用于指示所述被依赖任务或所述依赖任务处理完成；

[0147] 所述处理单元601还用于：根据所述通知消息，对所述事件的状态进行更新；若确定所述事件更新后的状态符合预设条件，则释放所述事件的标识。

[0148] 图7为本申请实施提供的又一种处理装置的结构示意图，该装置具有实现上述图3所示意的方法流程中第二处理装置所执行的行为的功能。所述处理装置700中包括：收发单元701和处理单元702；其中，收发单元701，用于接收第一处理装置发送的指令，所述指令中包括多个任务和所述多个任务的任务描述信息，所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系，其中，存在依赖关系的任意两个任务中，其中一个任务的处理需要等待另一个任务的处理结果；

[0149] 处理单元702，用于根据所述多个任务之间的依赖关系对所述多个任务进行处理。

[0150] 在一种可能的设计中，所述任务描述信息包括事件的标识，所述事件用于指示所述事件对应的任务之间的依赖关系，所述事件对应的任务为所述多个任务中位于不同的任务流且具有依赖关系的两个或两个以上的任务，所述两个或两个以上任务包括一个依赖任务以及一个或一个以上被依赖任务；所述依赖任务的处理依赖于所述各个被依赖任务的处理结果；

[0151] 所述处理单元702具体用于：

[0152] 确定所述各个被依赖任务均处理完成后，将所述事件的状态更新为就绪状态；

[0153] 确定所述依赖任务为待处理任务后，若检测到所述事件的状态为就绪状态，则对所述依赖任务进行处理。

[0154] 在一种可能的设计中，所述收发单元701还用于：

[0155] 确定所述被依赖任务或所述依赖任务处理完成后，向所述第一处理装置返回通知消息，所述通知消息用于指示所述被依赖任务或所述依赖任务处理完成。

[0156] 需要说明的是，本申请实施例中对单元的划分是示意性的，仅仅为一种逻辑功能划分，实际实现时可以有另外的划分方式。在本申请的实施例中的各功能单元可以集成在一个处理单元中，也可以是各个单元单独物理存在，也可以两个或两个以上单元集成在一个单元中。上述集成的单元既可以采用硬件的形式实现，也可以采用软件功能单元的形式实现。

[0157] 在上述实施例中，可以全部或部分地通过软件、硬件、固件或者其任意组合来实现。当使用软件实现时，可以全部或部分地以计算机程序产品的形式实现。所述计算机程序产品包括一个或多个计算机指令。在计算机上加载和执行所述计算机程序指令时，全部或

部分地产生按照本发明实施例所述的流程或功能。所述计算机可以是通用计算机、专用计算机、计算机网络、或者其他可编程装置。所述计算机指令可以存储在计算机可读存储介质中，或者从一个计算机可读存储介质向另一个计算机可读存储介质传输，例如，所述计算机指令可以从一个网站站点、计算机、服务器或数据中心通过有线(例如同轴电缆、光纤、数字用户线(DSL))或无线(例如红外、无线、微波等)方式向另一个网站站点、计算机、服务器或数据中心进行传输。所述计算机可读存储介质可以是计算机能够存取的任何可用介质或者是包含一个或多个可用介质集成的服务器、数据中心等数据存储设备。所述可用介质可以是磁性介质，(例如，软盘、硬盘、磁带)、光介质(例如，DVD)、或者半导体介质(例如固态硬盘Solid State Disk(SSD))等。

[0158] 本申请是参照本申请实施例的方法、装置(设备)和计算机程序产品的流程图和/或方框图来描述的。应理解可由计算机程序指令实现流程图和/或方框图中的每一流程和/或方框、以及流程图和/或方框图中的流程和/或方框的结合。可提供这些计算机程序指令到通用计算机、专用计算机、嵌入式处理机或其他可编程数据处理设备的处理器以产生一个机器，使得通过计算机或其他可编程数据处理设备的处理器执行的指令产生用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的装置。

[0159] 这些计算机程序指令也可存储在能引导计算机或其他可编程数据处理设备以特定方式工作的计算机可读存储器中，使得存储在该计算机可读存储器中的指令产生包括指令装置的制造品，该指令装置实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能。

[0160] 这些计算机程序指令也可装载到计算机或其他可编程数据处理设备上，使得在计算机或其他可编程设备上执行一系列操作步骤以产生计算机实现的处理，从而在计算机或其他可编程设备上执行的指令提供用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的步骤。

[0161] 尽管结合具体特征及其实施例对本申请进行了描述，显而易见的，在不脱离本申请的精神和范围的情况下，可对其进行各种修改和组合。相应地，本说明书和附图仅仅是所附权利要求所界定的本申请的示例性说明，且视为已覆盖本申请范围内的任意和所有修改、变化、组合或等同物。显然，本领域的技术人员可以对本申请进行各种改动和变型而不脱离本申请的精神和范围。这样，倘若本申请的这些修改和变型属于本申请权利要求及其等同技术的范围之内，则本申请也意图包含这些改动和变型在内。

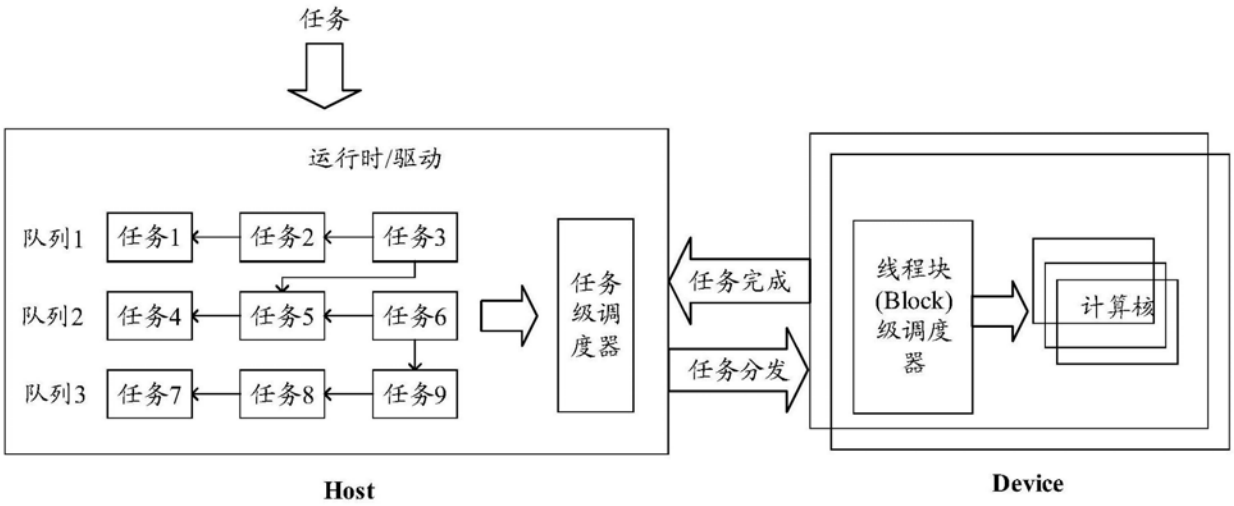

任务
运行时/驱动
队列1
任务1
任务2
任务3
队列2
任务4
任务5
任务6
队列3
任务7
任务8
任务9
任务级调度器
任务完成
任务分发
线程块(Block)级调度器
计算核
Host
Device

图1

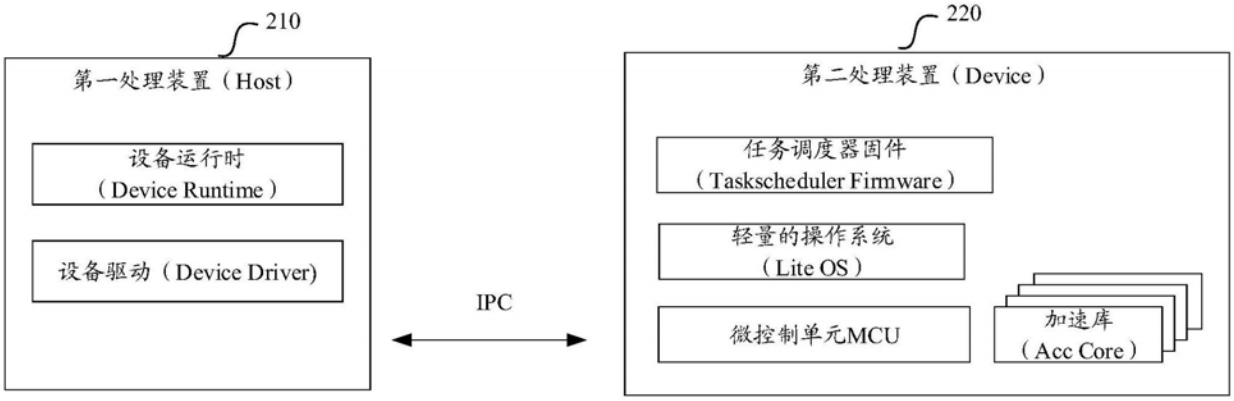

210
第一处理装置（Host）
设备运行时（Device Runtime）
设备驱动（Device Driver)
IPC
220
第二处理装置（Device）
任务调度器固件（Taskscheduler Firmware）
轻量的操作系统（Lite OS）
微控制单元MCU
加速库（Acc Core）

图2

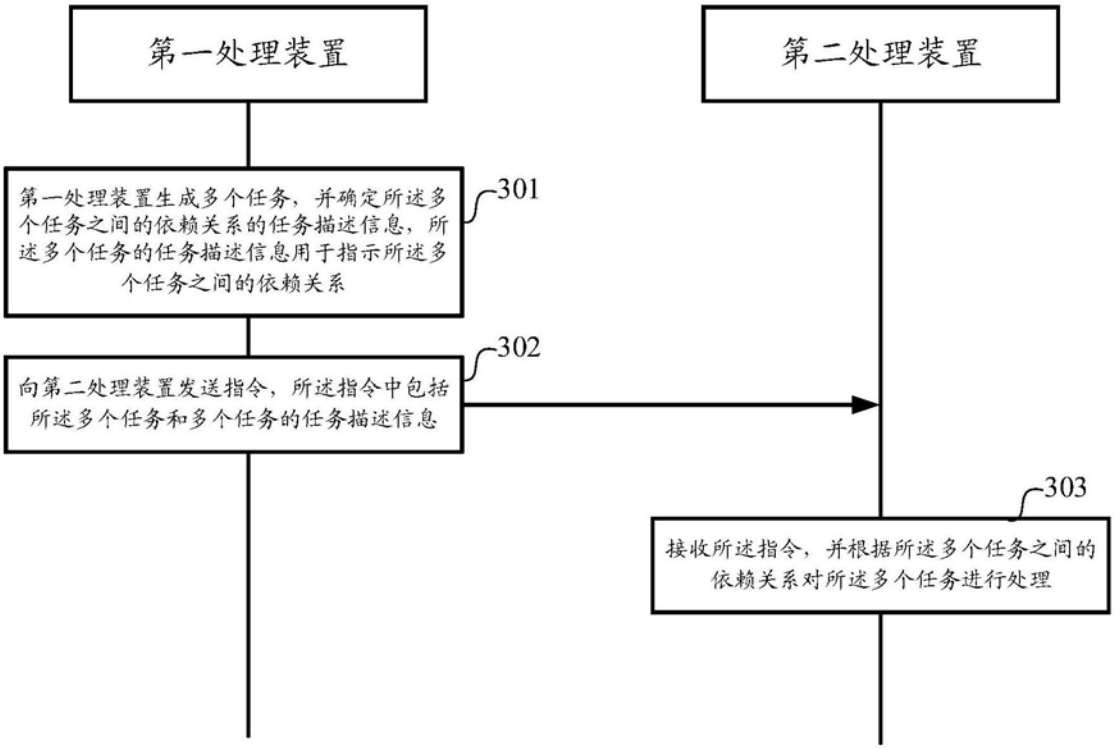

第一处理装置
第二处理装置
第一处理装置生成多个任务，并确定所述多个任务之间的依赖关系的任务描述信息，所述多个任务的任务描述信息用于指示所述多个任务之间的依赖关系
301
向第二处理装置发送指令，所述指令中包括所述多个任务和多个任务的任务描述信息
302
接收所述指令，并根据所述多个任务之间的依赖关系对所述多个任务进行处理
303

图3

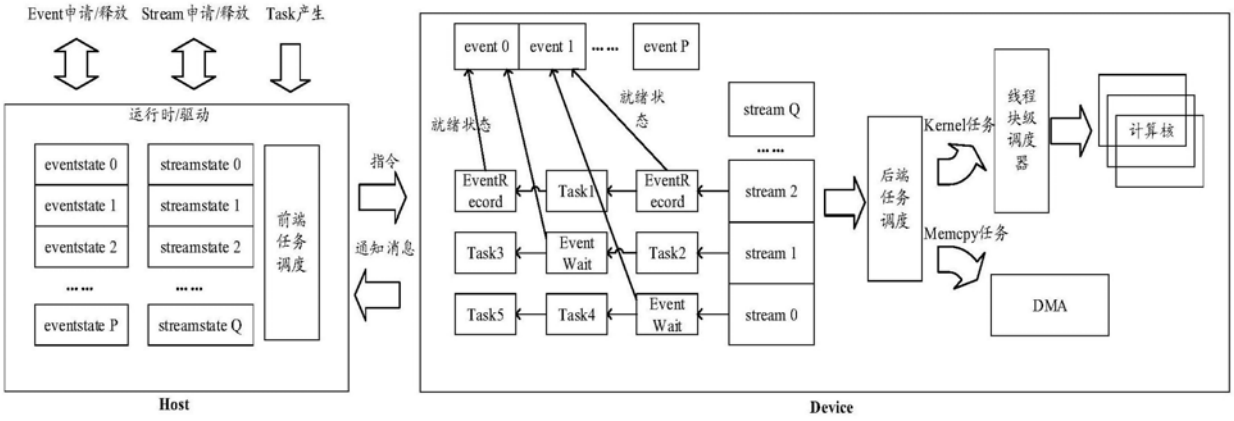

Event申请/释放
Stream申请/释放
Task产生
运行时/驱动
eventstate 0
eventstate 1
eventstate 2
eventstate P
streamstate 0
streamstate 1
streamstate 2
streamstate Q
前端任务调度
指令
通知消息
Host
event 0
event 1
event P
就绪状态
就绪状态
stream Q
stream 2
stream 1
stream 0
EventRecord
Task1
Task3
Task5
Task2
Task4
Event Wait
后端任务调度
Kernel任务
Memcpy任务
线程块级调度器
计算核
DMA
Device

图4

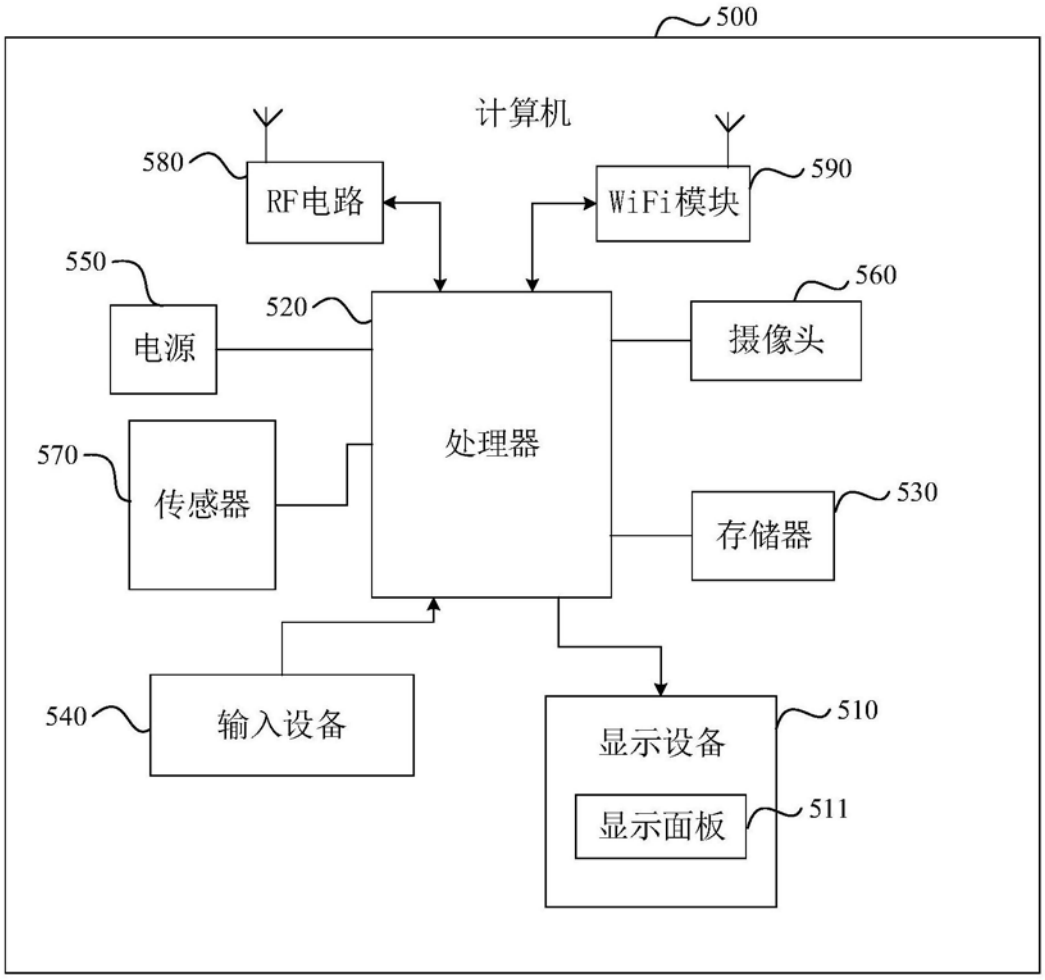

500
计算机
580
RF电路
WiFi模块
590
550
电源
520
处理器
560
摄像头
570
传感器
530
存储器
540
输入设备
显示设备
510
显示面板
511

图5

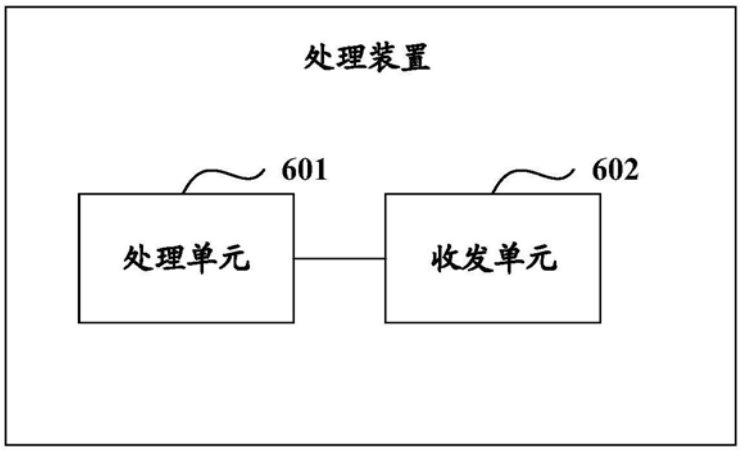

处理装置
601
处理单元
602
收发单元

图6

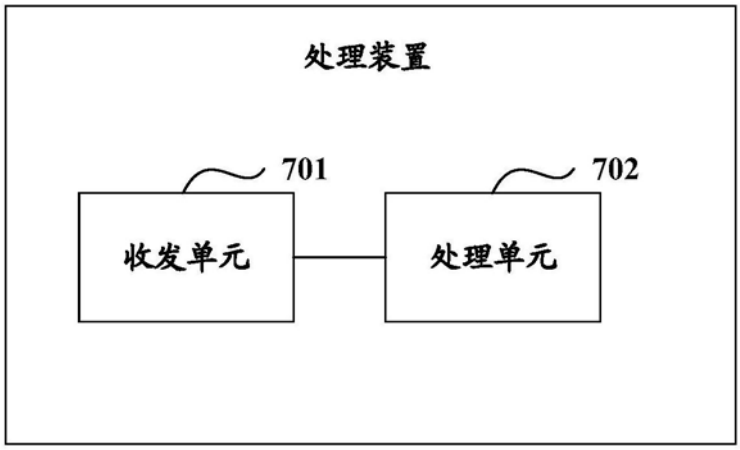


图7