(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 113536221 B
(45)授权公告日 2023.12.15

(21)申请号 202010318387.0

(22)申请日 2020.04.21

(65)同一申请的已公布的文献号
申请公布号 CN 113536221 A

(43)申请公布日 2021.10.22

(73)专利权人 中科寒武纪科技股份有限公司
地址 100191 北京市海淀区知春路7号致真大厦D座16层1601房

(72)发明人 请求不公布姓名 请求不公布姓名 请求不公布姓名

(74)专利代理机构 北京林达刘知识产权代理事务所(普通合伙) 11277
专利代理师 刘新宇

(51)Int.Cl.
G06F 17/16(2006.01)
G06F 9/30(2006.01)

(56)对比文件
CN 109284475 A,2019.01.29
CN 109213962 A,2019.01.15
CN 109190756 A,2019.01.11
CN 109240746 A,2019.01.18
CN 109992743 A,2019.07.09
CN 110647975 A,2020.01.03
EP 3343356 A1,2018.07.04
US 2019266217 A1,2019.08.29
宋宇鲲 等;.一种极低IO带宽需求的大维度矩阵链式矩阵乘法器设计.电子技术应用.2019,第45卷(第9期),32-38.
Victoria Erofeeva.Multi-Sensor Task Assignment Using Linear Matrix Inequalities in the Multiple Target Tracking Problem.Elsevier Science.2018,第51卷(第15期),880-885.
蒋川群;杜奕.稀疏矩阵相乘的一个改进算法.计算机工程与应用.2009,(第19期),全文.
刘仲;田希.面向多核向量处理器的矩阵乘法向量化方法.计算机学报.2017,(第10期),全文.

审查员 杨楚莹

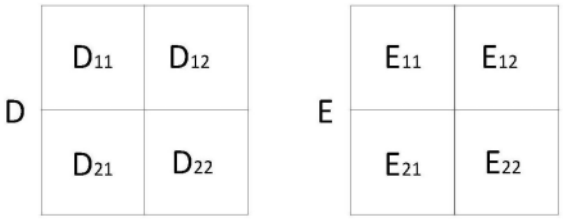
权利要求书3页 说明书20页 附图5页

(54)发明名称

运算方法、处理器以及相关产品

(57)摘要

本公开涉及运算方法、处理器以及相关产品。所述产品包括存储器件、接口装置和控制器件以及上述人工智能芯片;其中,所述人工智能芯片与所述存储器件、所述控制器件以及所述接口装置分别连接;所述存储器件,用于存储数据;所述接口装置,用于实现所述人工智能芯片与外部设备之间的数据传输;所述控制器件,用于对所述人工智能芯片的状态进行监控。通过以上运算方法或相关产品,本公开可以提高相关产品在进行矩阵乘法运算时的运算效率。



CN 113536221 B

1.一种基于处理元件矩阵的矩阵乘的运算方法,其特征在于,应用于处理器,所述处理器包括两个以上处理元件,所述两个以上处理元件以二维矩阵排列,处理元件包括至少一个寄存器,所述方法实现对第一矩阵和第二矩阵的矩阵乘法运算,

所述方法包括:

将第一矩阵加载到处理元件的寄存器中;

针对第二矩阵的每一行,将所述每一行中的元素与第一矩阵的每一列元素对应存储到处理元件的寄存器,与第一矩阵的每一列中的元素分别求乘积,计算一列乘积的和得到第一中间结果;或者,针对第二矩阵的每一列,将所述每一列中的元素与第一矩阵的每一行元素对应存储到处理元件的寄存器,与第一矩阵的每一行中的元素分别求乘积,计算一行乘积的和得到第一中间结果;

将第一中间结果进行处理得到第一矩阵和第二矩阵的乘积;

所述方法还包括:从输入矩阵中确定待加载矩阵;其中,输入矩阵包括左乘矩阵和右乘矩阵,待加载矩阵为左乘矩阵或右乘矩阵;根据处理元件的排列以及待加载矩阵的行秩以及列秩确定是否对待加载矩阵进行分块;若要对待加载矩阵进行分块,则根据待处理元件的排列以及待加载矩阵的行秩以及列秩对待加载矩阵进行分块得到两个以上第一矩阵,其中,通过直接对所述待加载矩阵沿着行方向或列方向进行划分,使得划分出的第一矩阵的行秩、列秩分别接近所述处理元件的行数、列数。

2.根据权利要求1所述的方法,其特征在于,第一矩阵为左乘矩阵、第二矩阵为右乘矩阵,

针对第二矩阵中的每一列元素,将该列元素中的每个元素与第一矩阵中对应的一列元素存储到处理元件的寄存器,控制每一个处理元件对相应的寄存器内的元素进行乘法运算得到元素乘积,计算每一行元素乘积的和得到第一中间结果,

其中,第一矩阵中与所述每个元素对应的一列元素是指,该元素在所述第二矩阵中的行数与一列元素的列数相同。

3.根据权利要求1所述的方法,其特征在于,第一矩阵为右乘矩阵、第二矩阵为左乘矩阵,

针对第二矩阵中的每一行元素,将该行元素中的每个元素与第一矩阵中对应的一行元素存储到处理元件的寄存器,控制每一个处理元件对相应的寄存器内的元素进行乘法运算得到元素乘积,计算每一列元素乘积的和得到第一中间结果,

其中,第一矩阵中与所述每个元素对应的一行元素是指,该元素在所述第二矩阵中的列数与一行元素所在的行数相同。

4.根据权利要求1-3任意一项所述的方法,其特征在于,所述方法还包括:

根据处理元件的排列,从输入矩阵中确定不需要进行分块的矩阵为第一矩阵,输入矩阵中的另一矩阵为第二矩阵,输入矩阵包括左乘矩阵和右乘矩阵。

5.根据权利要求1所述的方法,其特征在于,所述方法还包括:

根据对待加载矩阵分块的方式,对输入矩阵中除了待加载矩阵以外的另一个矩阵进行分块得到两个以上第二矩阵;

根据第一矩阵和对应的第二矩阵的乘积,按照矩阵乘的规则计算所述左乘矩阵和所述右乘矩阵的乘积。

6.根据权利要求1所述的方法,其特征在于,所述处理器包括多组寄存器,所述方法还包括:

在对所述输入矩阵进行分块后,在所述多组寄存器中堆叠存储所述两个以上第一矩阵,每组存储一个第一矩阵。

7.一种处理器,其特征在于,所述处理器包括两个以上处理元件,所述两个以上处理元件以二维矩阵排列,处理元件包括至少一个寄存器,所述处理器用于对第一矩阵和第二矩阵执行矩阵乘法运算,

所述处理器还包括控制器,所述控制器用于将第一矩阵加载到处理元件的寄存器中;

针对第二矩阵的每一行,所述控制器用于将所述每一行中的元素与第一矩阵的每一列元素对应存储到处理元件的寄存器,与第一矩阵的每一列中的元素分别求乘积,计算一列乘积的和得到第一中间结果;或者,针对第二矩阵的每一列,所述控制器用于将所述每一列中的元素与第一矩阵的每一行元素对应存储到处理元件的寄存器,与第一矩阵的每一行中的元素分别求乘积,计算一行乘积的和得到第一中间结果;

所述控制器还用于将第一中间结果进行处理得到第一矩阵和第二矩阵的乘积;

所述控制器还用于从输入矩阵中确定待加载矩阵;其中,输入矩阵包括左乘矩阵和右乘矩阵,待加载矩阵为左乘矩阵或右乘矩阵;根据处理元件的排列以及待加载矩阵的行秩以及列秩确定是否对待加载矩阵进行分块;若要对待加载矩阵进行分块,则所述控制器用于根据待处理元件的排列以及待加载矩阵的行秩以及列秩对待加载矩阵进行分块得到两个以上第一矩阵,其中,通过直接对所述待加载矩阵沿着行方向或列方向进行划分,使得划分出的第一矩阵的行秩、列秩分别接近所述处理元件的行数、列数。

8.根据权利要求7所述的处理器,其特征在于,第一矩阵为左乘矩阵、第二矩阵为右乘矩阵,

针对第二矩阵中的每一列元素,所述控制器用于将该列元素中的每个元素与第一矩阵中对应的一列元素存储到处理元件的寄存器,控制每一个处理元件对相应的寄存器内的元素进行乘法运算得到元素乘积,计算每一行元素乘积的和得到第一中间结果,

其中,第一矩阵中与所述每个元素对应的一列元素是指,该元素在所述第二矩阵中的行数与一列元素的列数相同。

9.根据权利要求7所述的处理器,其特征在于,第一矩阵为右乘矩阵、第二矩阵为左乘矩阵,

针对第二矩阵中的每一行元素,所述控制器用于将该行元素中的每个元素与第一矩阵中对应的一行元素存储到处理元件的寄存器,控制每一个处理元件对相应的寄存器内的元素进行乘法运算得到元素乘积,计算每一列元素乘积的和得到第一中间结果,

其中,第一矩阵中与所述每个元素对应的一行元素是指,该元素在所述第二矩阵中的列数与一行元素所在的行数相同。

10.根据权利要求7-9任意一项所述的处理器,其特征在于,所述处理器还用于根据处理元件的排列,从输入矩阵中确定不需要进行分块的矩阵为第一矩阵,输入矩阵中的另一矩阵为第二矩阵,输入矩阵包括左乘矩阵和右乘矩阵。

11.根据权利要求7所述的处理器,其特征在于,所述控制器还用于根据对待加载矩阵分块的方式,对输入矩阵中除了待加载矩阵以外的另一个矩阵进行分块得到两个以上第二

矩阵;根据第一矩阵和对应的第二矩阵的乘积,按照矩阵乘的规则计算所述左乘矩阵和所述右乘矩阵的乘积。

12.根据权利要求7所述的处理器,其特征在于,所述处理器包括多组寄存器,在对所述输入矩阵进行分块后,所述控制器还用于在所述多组寄存器中堆叠存储所述两个以上第一矩阵,每组存储一个第一矩阵。

13.一种人工智能芯片,其特征在于,所述芯片包括如权利要求7-12中任意一项所述的处理器。

14.一种电子设备,其特征在于,包括如权利要求13所述的人工智能芯片。

运算方法、处理器以及相关产品

技术领域

[0001] 本公开涉及信息处理技术领域,特别是涉及一种运算方法、处理器以及相关产品。

背景技术

[0002] 在人工智能技术领域,神经网络算法是最近非常流行的一种机器学习算法,在各种领域中都取得了非常好的效果,比如图像识别,语音识别,自然语言处理等。随着神经网络算法的发展,算法的复杂度也越来越高,为了提高识别度,模型的规模也在逐渐增大。用GPU和CPU处理起这些大规模的模型,要花费大量的计算时间,并且耗电量很大。

发明内容

[0003] 基于此,有必要针对上述技术问题,提供一种能够提高运算效率的运算方法、处理器及相关产品。

[0004] 根据本公开的一方面,提供了一种基于处理元件矩阵的矩阵乘的运算方法,应用于处理器,所述处理器包括两个以上处理元件,所述两个以上处理元件以二维矩阵排列,处理元件包括至少一个寄存器,所述方法实现对第一矩阵和第二矩阵的矩阵乘法运算,

[0005] 所述方法包括:

[0006] 将第一矩阵加载到处理元件的寄存器中,第一矩阵中的元素在矩阵中的排列方式和在处理元件的寄存器中的排列方式相同;

[0007] 针对第二矩阵的每一行,将所述每一行中的元素与第一矩阵的每一列元素对应存储到处理元件的寄存器,与第一矩阵的每一列中的元素分别求乘积,计算一列乘积的和得到第一中间结果;或者,针对第二矩阵的每一列,将所述每一列中的元素与第一矩阵的每一行元素对应存储到处理元件的寄存器,与第一矩阵的每一行中的元素分别求乘积,计算一行乘积的和得到第一中间结果;

[0008] 将第一中间结果进行处理得到第一矩阵和第二矩阵的乘积。

[0009] 根据本公开的另一方面,提供了一种处理器,所述处理器包括两个以上处理元件,所述两个以上处理元件以二维矩阵排列,处理元件包括至少一个寄存器,所述处理器用于对第一矩阵和第二矩阵执行矩阵乘法运算,

[0010] 所述处理器还包括控制器,所述控制器用于将第一矩阵加载到处理元件的寄存器中;

[0011] 针对第二矩阵的每一行,所述控制器用于将所述每一行中的元素与第一矩阵的每一列元素对应存储到处理元件的寄存器,与第一矩阵的每一列中的元素分别求乘积,计算一列乘积的和得到第一中间结果;或者,针对第二矩阵的每一列,所述控制器用于将所述每一列中的元素与第一矩阵的每一行元素对应存储到处理元件的寄存器,与第一矩阵的每一行中的元素分别求乘积,计算一行乘积的和得到第一中间结果;

[0012] 所述控制器还用于将第一中间结果进行处理得到第一矩阵和第二矩阵的乘积。

[0013] 根据本公开的另一方面,提供了一种人工智能芯片,所述芯片包括如上所述的处

理器。

[0014] 根据本公开的另一方面,提供了一种电子设备,包括如上所述的人工智能芯片。

[0015] 根据本公开的另一方面,提供了一种电子设备,包括如上所述的处理器。

[0016] 根据本公开上述各实施方式的矩阵乘的运算方法、处理器,更适用于以阵列排布的处理元件组成的处理器,运算效率高。且对于满足处理元件的排列的任意规模的输入矩阵,可以得到矩阵乘法的运算结果,可以减少访存次数,降低带宽压力,提高运算的效率。

[0017] 根据下面参考附图对示例性实施例的详细说明,本公开的其它特征及方面将变得清楚。

附图说明

[0018] 包含在说明书中并且构成说明书的一部分的附图与说明书一起示出了本公开的示例性实施例、特征和方面,并且用于解释本公开的原理。

[0019] 图1示出根据本公开一实施例的处理器的示意图。

[0020] 图2a和图2b分别示出了不同的划分方式的示例。

[0021] 图3示出根据本公开一实施例的运算方法的流程图。

[0022] 图4示出根据本公开一实施例的处理元件组成的阵列的示意图。

[0023] 图5示出根据本公开一实施例的分块的示意图。

[0024] 图6示出根据本公开一实施例的对矩阵划分的示例。

[0025] 图7示出根据本公开实施例的板卡的结构框图。

具体实施方式

[0026] 下面将结合本公开实施例中的附图,对本公开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公开一部分实施例,而不是全部的实施例。基于本公开中的实施例,本领域技术人员在没有做出创造性劳动前提下所获得的所有其他实施例,都属于本公开保护的范围。

[0027] 应当理解,本公开的权利要求、说明书及附图中的术语“第一”、“第二”、“第三”和“第四”等是用于区别不同对象,而不是用于描述特定顺序。本公开的说明书和权利要求书中使用的术语“包括”和“包含”指示所描述特征、整体、步骤、操作、元素和/或组件的存在,但并不排除一个或多个其它特征、整体、步骤、操作、元素、组件和/或其集合的存在或添加。

[0028] 还应当理解,在此本公开说明书中所使用的术语仅仅是出于描述特定实施例的目的,而并不意在限定本公开。如在本公开说明书和权利要求书中所使用的那样,除非上下文清楚地指明其它情况,否则单数形式的“一”、“一个”及“该”意在包括复数形式。还应当进一步理解,在本公开说明书和权利要求书中使用的术语“和/或”是指相关联列出的项中的一个或多个的任何处理以及所有可能处理,并且包括这些处理。

[0029] 如在本说明书和权利要求书中所使用的那样,术语“如果”可以依据上下文被解释为“当...时”或“一旦”或“响应于确定”或“响应于检测到”。类似地,短语“如果确定”或“如果检测到[所描述条件或事件]”可以依据上下文被解释为意指“一旦确定”或“响应于确定”或“一旦检测到[所描述条件或事件]”或“响应于检测到[所描述条件或事件]”。

[0030] 矩阵运算在利用人工智能对信息进行处理的过程中占据比较大的计算量,并且现

有的处理器在处理矩阵运算的过程中把矩阵运算拆解成乘法运算和加法运算,需要频繁的从内存中读取数据,运算的效率很低。

[0031] 相关技术中,对于输入矩阵规模比较大的矩阵乘法,为了提高矩阵运算的效率,通常采用多级流水线的方式实现运算的过程,但多级流水线由于每一级对输入数据中的一部分进行处理,因此,需要频繁的从内存中读取数据,频繁访问内存导致对带宽的要求较高。

[0032] 为了解决上述技术问题,本公开提供了一种运算方法以及执行该运算方法的处理器。处理器可以包括多个处理元件,在一些实施方式中,多个处理元件可以以二维矩阵的形式排列以更好的适应矩阵运算,每个处理元件可以包括至少一个寄存器。

[0033] 图1示出根据本公开一实施例的处理器的示意图。如图1所示,多个处理元件PE(Processing Element)以二维矩阵的形式排列,每个处理元件与相邻的处理元件之间连接,每个PE中可以设置有至少一个寄存器(register)(图中未示出)。处理器还可以包括控制器和存储器,其中,控制器和存储器都与多个处理元件连接,且控制器可以连接存储器。所述控制器用于从存储器中加载数据到处理元件的寄存器中,并控制处理元件对输入数据进行处理。

[0034] 在本公开的实施例的运算过程中,控制器可以先将一个矩阵的元素加载到各个PE对应的寄存器中,然后将另一个矩阵的元素按行或者按列或者按照元素遍历的方式根据加载到寄存器的矩阵中的元素加载的位置存储至对应的寄存器中,然后控制每个PE对PE内设置的寄存器存储的元素进行运算。

[0035] 在一种可能的实现方式中,存储器中还可以存储有可执行程序,可执行程序中可以包括指令,处理器执行指令可以实现矩阵乘法运算。控制器中可以设置有加载器、译码器等,其中,加载器可以用于将存储器中的输入数据加载到处理元件的寄存器中,译码器可以根据加载后输入数据的存储地址的变化对可执行程序中访问数据的指令进行译码,比如说,对于访问数据的指令,通过译码获得数据在寄存器中存储的地址赋值给访问数据的指令,并将译码后的指令发送给处理元件,由处理元件执行指令,从而实现对数据的处理。

[0036] 在一种可能的实现方式中,存储器可以为片上缓存,控制器可以将片外闪存上的可执行程序以及输入数据(例如,输入矩阵,包括左乘矩阵和右乘矩阵)加载到上述存储器(片上缓存)中,再进行之后的矩阵乘法运算的过程。

[0037] 在一种可能的实现方式中,控制器也可以直接从片外内存上加载输入矩阵以及可执行程序到处理元件的寄存器中,本公开对此不作限定。

[0038] PE中还可以包括运算器以完成指定的运算,以矩阵运算为例,PE中可以包括例如乘法器、加法器等,各个PE中的具体结构可以相同,也可以存在不同,本公开对此不作限定。PE中还可以包括其他类型的运算器,以适应各种不同的运算过程,本公开对PE包括的运算器的数量和类型不作限定。

[0039] 乘法操作的输入矩阵可以包括左乘矩阵和右乘矩阵,其中,左乘矩阵可以是指位于乘号左边的矩阵,右乘矩阵可以是指位于乘号右边的矩阵。

[0040] 本公开提供的运算方法用于实现对第一矩阵和第二矩阵的矩阵乘法运算。其中,在一个示例中,第一矩阵可以为左乘矩阵,第二矩阵可以为右乘矩阵;在另一个示例中,第一矩阵可以为右乘矩阵,第二矩阵可以为左乘矩阵。

[0041] 本公开的实施方式中,控制器可以将输入矩阵中的一个矩阵确定为待加载矩阵。

由于处理器中PE的数量以及排列方式是固定的,因此,在一些情况下控制器可以对待加载矩阵进行分块,在一些情况下,可以不对加载到处理器中的矩阵进行分块。对于输入矩阵中除了待加载矩阵以外的另一矩阵,可以不进行分块处理。

[0042] 在一种可能的实现方式中,控制器可以从输入矩阵中确定待加载矩阵,根据处理元件的排列以及待加载矩阵的行数和列数确定是否对待加载矩阵进行分块。其中,处理元件的排列可以是指处理元件的行数和列数,待加载矩阵的行秩、列秩可以是指该矩阵的行数和列数。待加载矩阵可以是左乘矩阵,也可以是右乘矩阵,本公开对此不作限定。

[0043] 若待加载矩阵的行数不大于处理元件的行数、且待加载矩阵的列数不大于处理元件的列数,则控制器可以不对待加载矩阵进行分块,若待加载矩阵的行数大于处理元件的行数,或者待加载矩阵的列数大于处理元件的列数,则控制器可以对待加载矩阵进行分块。

[0044] 在一种可能的实现方式中,在从输入矩阵中确定待加载矩阵时,控制器可以随机确定,也可以根据处理元件的排列优先确定不需要进行分块的矩阵为待加载矩阵,本公开对具体的确定方式不作限定。

[0045] 比如说,假设处理元件组成的阵列可以表示为PE_{MN},表示处理元件为M×N的矩阵,其中,M表示处理元件的行数、N表示处理元件的列数,M和N都为大于0的正整数。假设左乘矩阵为a_{mn},表示左乘矩阵为m×n的矩阵,其中,m表示矩阵a_{mn}的行数,n表示矩阵a_{mn}的列数,m和n都为正整数,右乘矩阵为b_{nk},表示右乘矩阵为n×k的矩阵,其中n为矩阵b_{nk}的行数,k为矩阵b_{nk}的列数,k为正整数。如果m小于M、n小于N,n大于M或者k大于N,那么控制器可以优选矩阵a_{mn}为待加载矩阵。

[0046] 在一种可能的实现方式中,若两个输入矩阵都满足不需要分块的条件,即都可以作为待加载矩阵,此时控制器可以随机确定其中一个为待加载矩阵,也可以选择包含元素较多的矩阵作为待加载矩阵,这样可以减少加载元素的次数,提高运算效率。

[0047] 若要对待加载矩阵进行分块,则控制器可以根据待处理元件的排列以及待加载矩阵的行秩以及列秩对待加载矩阵进行分块得到两个以上第一矩阵。

[0048] 需要说明的是,本公开的示例中以加载第一矩阵到各处理元件为例,也就是将待加载矩阵作为第一矩阵或者将对待加载矩阵分块后得到的矩阵作为第一矩阵。

[0049] 对于不需要分块的情况,如果加载的第一矩阵为左乘矩阵,那么控制器可以将右乘作为第二矩阵,如果加载的第一矩阵为右乘矩阵,那么控制器可以将左乘矩阵作为第二矩阵。

[0050] 对于需要分块的情况,如果对待加载矩阵进行分块得到两个以上第一矩阵,那么控制器可以根据情况对输入矩阵中的另一个矩阵进行处理。

[0051] 如果处理元件包括的寄存器无法存储全部的第一矩阵,这时,根据对待加载矩阵分块的方式的不同,控制器可以对输入矩阵中待加载矩阵以外的另一个矩阵进行分块,也可以不进行分块。

[0052] 比如说,若待加载矩阵为左乘矩阵,对待加载矩阵在行方向进行了分块,此时控制器可以不对另一个矩阵进行分块;如果待加载矩阵为左乘矩阵,对待加载矩阵在列方向进行了分块,此时控制器可以根据对待加载矩阵分块的方式,将输入矩阵中待加载矩阵以外的另一个矩阵进行分块得到两个以上第二矩阵。

[0053] 若待加载矩阵为右乘矩阵,对待加载矩阵在行方向进行了分块,此时控制器可以

根据对待加载矩阵分块的方式,将输入矩阵中待加载矩阵以外的另一个矩阵进行分块得到两个以上第二矩阵;如果待加载矩阵为右乘矩阵,对待加载矩阵在列方向进行了分块,此时控制器可以不对另一个矩阵进行分块。

[0054] 如果待加载矩阵为a_{mn},那么根据矩阵a_{mn}的行数和列数以及处理元件的行数和列数确定是否需要对矩阵a_{mn}进行分块,如果矩阵a_{mn}的行数m不大于处理元件的行数M、且列数n不大于处理元件的列数N,则可以不对矩阵a_{mn}进行分块。如果矩阵a_{mn}的行数m大于处理元件的行数M、或者列数n大于处理元件的列数N,则可以对矩阵a_{mn}在行方向或者列方向进行分块。

[0055] 如果待加载矩阵为b_{nk},那么根据矩阵b_{nk}的行数和列数以及处理元件的行数和列数确定是否需要对矩阵b_{nk}进行分块,如果矩阵b_{nk}的行数n不大于处理元件的行数M、且列数k不大于处理元件的列数N,则可以不对矩阵b_{nk}进行分块。如果矩阵b_{nk}的行数n大于处理元件的行数M、或列数k大于处理元件的列数N,则可以对矩阵b_{nk}在行方向或者列方向进行分块。

[0056] 在一种可能的实现方式中,分块后得到的矩阵满足不需要再进行分块的条件,也就是说,分块后矩阵的行数不大于处理元件的行数、且列数不大于处理元件的列数。

[0057] 如果矩阵a_{mn}的行数m大于处理元件的行数M、列数n不大于处理元件的列数N,则控制器可以对矩阵a_{mn}在行方向进行分块,由于矩阵a_{mn}为左乘矩阵,因此在行方向进行分块,并不影响与右乘矩阵的正常的运算,因此控制器可以不对右乘矩阵进行分块处理。如果矩阵a_{mn}的行数m不大于处理元件的行数M、列数n大于处理元件的列数N,则可以对矩阵a_{mn}在列方向进行分块,此时,控制器可以根据对矩阵a_{mn}在列方向进行分块的方式对右乘矩阵的行方向进行分块,对左乘矩阵列方向和右乘矩阵行方向以相同的方式进行分块,所述相同的方式分块指的是分块后所得的第一矩阵的列数和第二矩阵的行数是相同的,以保证能正常完成矩阵运算。如果矩阵a_{mn}的行数m大于处理元件的行数M、列数n大于处理元件的列数N,则控制器可以对矩阵a_{mn}在行方向和列方向进行分块,可以根据对矩阵a_{mn}在列方向进行分块的方式对右乘矩阵的行方向进行分块,对左乘矩阵列方向和右乘矩阵行方向以相同的方式进行分块,所述相同的方式分块指的是分块后所得的第一矩阵的列数和第二矩阵的行数是相同的,以保证能正常完成矩阵运算。

[0058] 如果矩阵b_{nk}的行数n不大于处理元件的行数M、列数k大于处理元件的列数N,则控制器可以对矩阵b_{nk}在列方向进行分块。由于矩阵b_{nk}为右乘矩阵,因此在列方向进行分块并不影响与左乘矩阵的正常的运算,因此控制器可以不对左乘矩阵进行分块处理。如果矩阵b_{nk}的行数n大于处理元件的行数M、列数k不大于处理元件的列数N,则可以对矩阵b_{nk}在行方向进行分块,此时,控制器可以根据对矩阵b_{nk}在行方向进行分块的方式对左乘矩阵的列方向进行分块,对左乘矩阵列方向和右乘矩阵行方向以相同的方式进行分块,所述相同的方式分块指的是分块后所得的第一矩阵的列数和第二矩阵的行数是相同的,以保证能正常完成矩阵运算。如果矩阵b_{nk}的行数n大于处理元件的行数M、列数k大于处理元件的列数N,则控制器可以对矩阵b_{nk}在行方向和列方向进行分块,此时,控制器可以根据对矩阵b_{nk}在行方向进行分块的方式对左乘矩阵的列方向进行分块,对左乘矩阵列方向和右乘矩阵行方向以相同的方式进行分块,所述相同的方式分块指的是分块后所得的第一矩阵的列数和第二矩阵的行数是相同的,以保证能正常完成矩阵运算。

[0059] 在一种可能的实现方式中,可以按照分块后的矩阵的行秩和列秩尽量接近处理元件的行数和列数的方式进行分块,这样可以提高运算的效率,缩短运算时间。也就是说,假设处理元件为4×4的阵列,那么可以先按照分块后的矩阵为4×4的方式进行分块,这样可以最大效率的利用处理元件,提高运算效率。

[0060] 举例来说,假设处理元件为2×2的阵列,左乘矩阵为2×4矩阵、右乘矩阵为4×3矩阵,这种情况下不管是加载左乘矩阵还是右乘矩阵,都需要对两者进行分块。分块的方式可以有很多种,图2a和图2b分别示出了多种不同的分块方式,矩阵a_{24}在列方向和矩阵b_{43}在行方向以相同的方式进行分块。图2a是分块的一个示例,矩阵a_{24}在列方向划分为两部分,每一部分包括两列,矩阵b_{43}在行方向划分为两部分,每一部分包括两行;图2b是分块的另一个示例,矩阵a_{24}在列方向划分为三部分,其中一部分包括两列、另外两部分都包括一列,矩阵b_{43}在行方向划分为三部分,其中一部分包括两行、另外两部分都包括一行。以上处理元件的排列以及输入矩阵的分块方式仅仅是本公开的一个示例,不以任何方式限制本公开。

[0061] 图2a中的分块方式划分出的矩阵的行秩和列秩更接近处理元件的行数和列数,这样,能够有助于提高处理元件的利用率,并且降低控制复杂度,对于相同的输入矩阵,由于分块后的块数较少,因此加载数据的次数少,这种分块方式运算的效率更高。

[0062] 对于左乘矩阵的行方向和右乘矩阵的列方向的分块方式,本公开不作具体的限定,只要分块后的矩阵都满足不需要再进行分块的条件即可。

[0063] 在一种可能的实现方式中,如果处理元件包含的寄存器的数量可以满足存储输入矩阵的需求,那么还可以采用堆叠存储的方式将划分后的第一矩阵存储到处理元件的寄存器中,来实现输入矩阵的乘法运算。比如说,每个处理元件可以包括多个寄存器,控制器可以把处理元件中的寄存器分为多个不同的组,控制器在对所述输入矩阵进行分块后,可以在多组寄存器中堆叠存储所述两个以上第一矩阵,每组存储一个第一矩阵。在该实施方式中,控制器可以将输入矩阵中待加载矩阵以外的另一个矩阵作为第二矩阵。需要说明的是,堆叠存储仅仅是一种可选的实现方式,本公开不限于此。

[0064] 图3示出根据本公开一实施例的运算方法的流程图。以不需要对待加载矩阵进行分块为例,先对本公开的运算方法进行说明,假设待加载矩阵为第一矩阵,输入矩阵中除了待加载矩阵以外的另一个矩阵为第二矩阵,如图3所示,本公开提供的运算方法可以包括以下步骤:

[0065] 步骤S11,将第一矩阵加载到各处理元件的寄存器中;

[0066] 在一种可能的实现方式中,第一矩阵中的元素在矩阵中的排列方式和在处理元件的寄存器中的排列方式相同;

[0067] 步骤S12,针对第二矩阵的每一行或者每一列,将所述每一行或者每一列中的元素与第一矩阵的每一列或每一行元素对应存储到处理元件的寄存器,与第一矩阵的每一列或每一行中的元素分别求乘积,计算一列或一行乘积的和得到第一中间结果;也就是说,针对第一矩阵的每一行或者每一列,将每一行或者每一列的元素存储到第一矩阵的每一列或者每一行元素存储的寄存器所在的处理元件的寄存器中。

[0068] 也就是说,针对第二矩阵的每一行,将所述每一行中的元素与第一矩阵的每一列元素对应存储到处理元件的寄存器,与第一矩阵的每一列中的元素分别求乘积,计算一列乘积的和得到第一中间结果;或者,针对第二矩阵的每一列,将所述每一列中的元素与第一

矩阵的每一行元素对应存储到处理元件的寄存器，与第一矩阵的每一行中的元素分别求乘积，计算一行乘积的和得到第一中间结果。

[0069] 步骤S13，将第一中间结果进行处理得到第一矩阵和第二矩阵的乘积。

[0070] 对于不分块的情况，控制器可以直接把左乘矩阵作为第一矩阵、右乘矩阵作为第二矩阵，或者将左乘矩阵作为第二矩阵、右乘矩阵作为第一矩阵，本公开对此不作限定。

[0071] 在一个示例中，第一矩阵为左乘矩阵，第二矩阵为右乘矩阵，那么在步骤S12中，针对第二矩阵中的每一列元素，可以将该列元素中的每个元素与第一矩阵中对应的一列元素存储到处理元件的寄存器(或者说，将该列元素中的每个元素存储至第一矩阵中对应的一列元素存储的寄存器所在的处理元件的寄存器中)，控制每一个处理元件对相应的寄存器内的元素进行乘法运算得到元素乘积，计算每一行元素乘积的和得到第一中间结果。其中，第一矩阵中与所述每个元素对应的一列元素是指，该元素在所述第二矩阵中的行数与一列元素在第二矩阵中的列数相同。

[0072] 在另一个示例中，第一矩阵为右乘矩阵，第二矩阵为左乘矩阵，那么在步骤S12中，针对第二矩阵中的每一行元素，可以将该行元素中的每个元素与第一矩阵中对应的一行元素存储到处理元件的寄存器，控制每一个处理元件对相应的寄存器内的元素进行乘法运算得到元素乘积，计算每一列元素乘积的和得到第一中间结果。其中，第一矩阵中与所述每个元素对应的一行元素是指，该元素在所述第二矩阵中的列数与一行元素所在的行数相同。

[0073] 根据加载到处理器中的矩阵为左乘矩阵或者右乘矩阵，步骤S13中对第一中间结果的处理方式不同。具体地，若第一矩阵为左乘矩阵，那么，得到的第一中间结果作为第一矩阵和第二矩阵的乘积矩阵的一列元素，第一中间结果在乘积矩阵中的列数与进行运算得到第一中间结果的第二矩阵中的列的列数相同；若第一矩阵为右乘矩阵，那么，得到的第一中间结果作为第一矩阵和第二矩阵的乘积矩阵的一行元素，第一中间结果在乘积矩阵中的行数与进行运算得到第一中间结果的第二矩阵中的行的行数相同。

[0074] 在一种可能的实现方式中，对于同一行或者同一列的处理元件，控制器可以控制该行或者该列的处理元件将每次计算得到的元素乘积移动到该行或者该列的一个处理元件中，并控制该行或者该列的一个处理元件计算元素乘积的和得到第一中间结果。比如说，在第一矩阵为左乘矩阵、第二矩阵为右乘矩阵时，在每次计算得到元素乘积时，控制器可以控制同一行的处理元件将计算得到的元素乘积移动到该行的一个处理元件中，并控制该一个处理元件计算元素乘积的和得到第一中间结果；在第一矩阵为右乘矩阵、第二矩阵为左乘矩阵时，在每次计算得到元素乘积时，控制器可以控制同一列的处理元件将计算得到的元素乘积移动到该列的一个处理元件中，并控制该一个处理元件计算元素乘积的和得到第一中间结果。其中，处理元件可以采用加法器计算元素乘积的和。其中的一个处理元件可以是存储有第一矩阵的元素的处理元件，也可以是未存储第一矩阵的元素的处理元件，本公开对此不作限定。

[0075] 以上示例仅仅是计算第一中间结果的一种方式，本公开不限于此，比如，还可以在处理元件阵列的行或者列上设置专门的加法器用于实现上述计算过程。

[0076] 示例1第一矩阵为左乘矩阵，第二矩阵为右乘矩阵

[0077] 假设第一矩阵a_{mn}和第二矩阵b_{nk}都为3×3矩阵，处理元件为4×4的阵列。

[0078] 图4示出根据本公开一实施例的处理元件组成的阵列的示意图。结合图4以及图3

对本公开的运算方法进行说明。

[0079] 假设第一矩阵$a_{33}=\begin{bmatrix}A_{11} & A_{12} & A_{13}\\ A_{21} & A_{22} & A_{23}\\ A_{31} & A_{32} & A_{33}\end{bmatrix}$，第二矩阵$b_{33}=\begin{bmatrix}B_{11} & B_{12} & B_{13}\\ B_{21} & B_{22} & B_{23}\\ B_{31} & B_{32} & B_{33}\end{bmatrix}$。

[0080] 将第一矩阵加载到所述处理元件的寄存器中，可以按照第一矩阵的行和列的排列方式加载到所述处理元件的寄存器中，也就是说，第一矩阵中的元素在矩阵中的排列方式和在处理元件的寄存器中的排列方式相同，换言之，所述排列方式相同指的是矩阵中所有元素的行下标与其所处的处理元件的行差值相同、所有元素的列下标与其所处的处理元件的列下标的差值相同。

[0081] 在一种可能的实现方式中，第一矩阵中的元素在矩阵中的行列数与加载有该元素的处理元件在处理元件组成的阵列中的行列数相同。

[0082] 举例来说，在一个示例中，控制器可以将A_{11}加载到PE_{11}的寄存器中、A_{12}加载到PE_{12}的寄存器中、A_{13}加载到PE_{13}的寄存器中、A_{21}加载到PE_{21}的寄存器中…A_{33}加载到PE_{33}的寄存器中，也就是说，第一矩阵中元素的下标可以与其所处的处理元件的下标完全相同，上述行下标差值和列下标差值都为0。

[0083] 在另一个示例中，控制器可以将A_{11}加载到PE_{12}的寄存器中、A_{12}加载到PE_{13}的寄存器中、A_{13}加载到PE_{14}的寄存器中、A_{21}加载到PE_{22}的寄存器中…A_{33}加载到PE_{34}的寄存器中，也就是说，第一矩阵中的元素在矩阵中的排列方式和在处理元件的寄存器中的排列方式相同，行下标差值为0、列下标的差值为1。

[0084] 需要说明的是，以上两个示例仅仅是加载第一矩阵的一些示例，不以任何方式限制本公开，本领域技术人员应当知道，只要满足第一矩阵中的元素在矩阵中的排列方式和在处理元件的寄存器中的排列方式相同即可。

[0085] 在一种可能的实现方式中，在加载完输入矩阵之后，对于步骤S12，控制器可以将第二矩阵的第一列中的元素B_{11}到第一矩阵中对应的一列元素存储到处理元件的寄存器，对应的一列元素是指该元素在所述第二矩阵中的行数与一列元素在第一矩阵中的列数相同，B_{11}在第一矩阵为第一行，那么对应的一列元素是指第一矩阵中的第一列元素。也就是说，控制器将元素B_{11}存储至A_{11}、A_{21}、A_{31}存储的寄存器所在的处理元件的寄存器中。

[0086] 控制器将第二矩阵的第一列中的元素B_{21}存储至A_{12}、A_{22}、A_{32}存储的寄存器所在的处理元件的寄存器中，将第二矩阵的第一列中的元素B_{31}存储至A_{13}、A_{23}、A_{33}存储的寄存器所在的处理元件的寄存器中。

[0087] 也就是说，B_{11}和A_{11}存储在同一个处理元件的寄存器中，B_{11}和A_{21}存储在同一个处理元件的寄存器中，B_{11}和A_{31}存储在同一个处理元件的寄存器中。B_{21}和A_{12}存储在同一个处理元件的寄存器中，B_{21}和A_{22}存储在同一个处理元件的寄存器中，B_{21}和A_{32}存储在同一个处理元件的寄存器中。B_{31}和A_{13}存储在同一个处理元件的寄存器中，B_{31}和A_{23}存储在同一个处理元件的寄存器中，B_{31}和A_{33}存储在同一个处理元件的寄存器中。

[0088] 处理器中的控制器控制处理元件分别对对应的寄存器内存储的元素求乘积，然后计算每一行乘积的和得到第一中间结果分别为：$B_{11}\times A_{11}+B_{21}\times A_{12}+B_{31}\times A_{13}$、$B_{11}\times A_{21}+B_{21}\times A_{22}+B_{31}\times A_{23}$、$B_{11}\times A_{31}+B_{21}\times A_{32}+B_{31}\times A_{33}$。假设第一矩阵和第二矩阵相乘得到的矩阵为$C_{33}$，那么上述第一中间结果可以表示为：$C_{11}$、$C_{21}$、$C_{31}$。

[0089] 在一种可能的实现方式中,示例性的,控制器可以将A_{11}加载到PE_{11}的寄存器中、A_{12}加载到PE_{12}的寄存器中、A_{13}加载到PE_{13}的寄存器中、A_{21}加载到PE_{21}的寄存器中…A_{33}加载到PE_{33}的寄存器中,也就是说,第一矩阵中元素的下标可以与其所处的处理元件的下标完全相同,上述行下标差值和列下标差值都为0。在本示例中,控制器将第二矩阵的第一列元素B_{11}、B_{21}、B_{31}存储至处理元件的寄存器之后,控制器控制处理元件采用乘法器对各自的寄存器中的元素求乘积得到元素乘积,控制器可以控制每一行处理元件将计算得到的元素乘积移动到该行的一个处理元件中,比如说,控制器可以控制PE_{11}、PE_{12}和PE_{13}将计算得到的元素乘积$B_{11}\times A_{11}$、$B_{21}\times A_{12}$、$B_{31}\times A_{13}$移动到处理元件PE_{14}中,控制PE_{14}采用加法器对上述元素乘积求和得到C_{11},需要说明的是,控制器也可以控制第一行的处理元件将元素乘积移动到PE_{11}、PE_{12}或者PE_{13}中,本公开对此不作限定。控制器控制第二行和第三行的处理元件执行类似的操作后,可以得到第一中间结果C_{11}、C_{21}、C_{31}。

[0090] 针对第二矩阵中的每一列,重复以上过程可以得到第一中间结果:C_{12}、C_{22}、C_{32}和C_{13}、C_{23}、C_{33}。利用上述第一中间结果即可得到第一矩阵和第二矩阵的乘积$c_{33}=\begin{bmatrix}C_{11} & C_{12} & C_{13}\\ C_{21} & C_{22} & C_{23}\\ C_{31} & C_{32} & C_{33}\end{bmatrix}$。

[0091] 在一种可能的实现方式中,对于得到的第一中间结果,可以按列存储即可得到第一矩阵和第二矩阵的乘积。也就是如上文所述的,第一矩阵为左乘矩阵时,以每一次得到的第一中间结果作为第一矩阵和第二矩阵的乘积矩阵的一列元素。第一中间结果在乘积矩阵中的列数与进行运算得到第一中间结果的第二矩阵中的列的列数相同是指,以上述示例为例,第二矩阵中的第一列元素与第一矩阵中的元素进行运算得到的第一中间结果C_{11}、C_{21}、C_{31}为c_{33}的第一列。

[0092] 示例2第一矩阵为右乘矩阵,第二矩阵为左乘矩阵

[0093] 仍然假设第一矩阵a_{mn}和第二矩阵b_{nk}都为3×3矩阵,处理元件为4×4的阵列。

[0094] 假设第一矩阵$a_{33}=\begin{bmatrix}A_{11} & A_{12} & A_{13}\\ A_{21} & A_{22} & A_{23}\\ A_{31} & A_{32} & A_{33}\end{bmatrix}$,第二矩阵$b_{33}=\begin{bmatrix}B_{11} & B_{12} & B_{13}\\ B_{21} & B_{22} & B_{23}\\ B_{31} & B_{32} & B_{33}\end{bmatrix}$。

[0095] 将第一矩阵加载到所输出处理元件的寄存器中,加载的方式可以参见示例1中加载第一矩阵的方式,不再赘述。

[0096] 在加载完第一矩阵之后,对于步骤S12,将第二矩阵的第一行中的元素B_{11}与第一矩阵中对应的一行元素存储到处理元件的寄存器,对应的一行元素是指该元素在所述第二矩阵中的列数与一列元素在第一矩阵中的行数相同,B_{11}在第一矩阵为第一列,那么对应的一列元素是指第一矩阵中的第一行元素。也就是说,控制器可以将元素B_{11}存储至A_{11}、A_{12}、A_{13}存储的寄存器所在的处理元件的寄存器中。

[0097] 将第二矩阵的第一行中的元素B_{12}存储至A_{21}、A_{22}、A_{23}存储的寄存器所在的处理元件的寄存器中,将第二矩阵的第一行中的元素B_{13}存储至A_{31}、A_{32}、A_{33}存储的寄存器所在的处理元件的寄存器中。

[0098] 也就是说,B_{11}和A_{11}存储在同一个处理元件的寄存器中,B_{11}和A_{12}存储在同一个处理元件的寄存器中,B_{11}和A_{13}存储在同一个处理元件的寄存器中。B_{12}和A_{21}存储在同一个处理元件的寄存器中,B_{12}和A_{22}存储在同一个处理元件的寄存器中,B_{12}和A_{23}存储在同一个处理元件

的寄存器中。B_{13}和A_{31}存储在同一个处理元件的寄存器中，B_{13}和A_{32}存储在同一个处理元件的寄存器中，B_{13}和A_{33}存储在同一个处理元件的寄存器中。

[0099] 处理器中的控制器控制处理元件分别对对应的寄存器内存储的元素求乘积，然后计算每一列乘积的和得到第一中间结果分别为：$B_{11}\times A_{11}+B_{12}\times A_{21}+B_{13}\times A_{31}$、$B_{11}\times A_{12}+B_{12}\times A_{22}+B_{13}\times A_{32}$、$B_{11}\times A_{13}+B_{12}\times A_{23}+B_{13}\times A_{33}$。假设第一矩阵和第二矩阵相乘得到的矩阵为$C_{33}$，那么上述第一中间结果可以表示为：$C_{11}$、$C_{12}$、$C_{13}$。

[0100] 在一种可能的实现方式中，示例性的，控制器可以将A_{11}加载到PE_{11}的寄存器中、A_{12}加载到PE_{12}的寄存器中、A_{13}加载到PE_{13}的寄存器中、A_{21}加载到PE_{21}的寄存器中…A_{33}加载到PE_{33}的寄存器中，也就是说，第一矩阵中元素的下标可以与其所处的处理元件的下标完全相同，上述行下标差值和列下标差值都为0。在本示例中，控制器将第二矩阵的第一行元素B_{11}、B_{12}、B_{13}存储至处理元件的寄存器之后，控制器控制处理元件采用乘法器对各自的寄存器中的元素求乘积得到元素乘积，控制器可以控制每一列处理元件将计算得到的元素乘积移动到该列的一个处理元件中，比如说，控制器可以控制PE_{11}、PE_{21}和PE_{31}将计算得到的元素乘积$B_{11}\times A_{11}$、$B_{12}\times A_{21}$、$B_{13}\times A_{31}$移动到处理元件PE_{41}中，控制PE_{14}采用加法器对上述元素乘积求和得到C_{11}，需要说明的是，控制器也可以控制第一行的处理元件将元素乘积移动到PE_{11}、PE_{21}或者PE_{31}中，本公开对此不作限定。控制器控制第二行和第三行的处理元件执行类似的操作后，可以得到第一中间结果C_{11}、C_{12}、C_{13}。

[0101] 针对第二矩阵中的每一行，重复以上过程可以得到第一中间结果：C_{21}、C_{22}、C_{23}和C_{31}、C_{32}、C_{33}。利用上述第一中间结果即可得到第一矩阵和第二矩阵的乘积$c_{33}=\begin{bmatrix}C_{11} & C_{12} & C_{13}\\ C_{21} & C_{22} & C_{23}\\ C_{31} & C_{32} & C_{33}\end{bmatrix}$。

[0102] 在一种可能的实现方式中，对于得到的第一中间结果，可以按列存储即可得到第一矩阵和第二矩阵的乘积。

[0103] 需要说明的是，以上示例中的处理元件的排列、输入矩阵等仅仅是为了清楚说明本公开运算方法的过程，不以任何方式限制本公开。

[0104] 根据本公开上述各实施方式的矩阵乘的运算方法，对于满足处理元件的排列的任意规模的输入矩阵，可以得到矩阵乘法的运算结果。

[0105] 对于不进行分块的情况，根据上述示例可以直接得到矩阵乘的结果。

[0106] 根据本公开上述各实施方式的矩阵乘的运算方法，更适用于以阵列排布的处理元件组成的处理器，相比于相关技术中的矩阵乘运算可以减少访存次数，降低带宽压力，提高运算的效率。对于需要进行分块的情况，对于分块后的第一矩阵和第二矩阵(可以是分块得到的，也可以是直接将另一个矩阵作为第二矩阵)，根据第一矩阵和对应的第二矩阵的乘积，按照矩阵乘的规则计算所述左乘矩阵和所述右乘矩阵的乘积。也就是说，可以将分块后得到的第一矩阵和第二矩阵作为矩阵的一个元素，按照矩阵乘的规则执行矩阵乘法的运算过程得到第二中间结果，根据第二中间结果进行计算可以得到所述输入矩阵的乘积。

[0107] 图5示出根据本公开一实施例的分块的示意图。如图5所示，将矩阵D和E按照以上所述的方式进行分块得到第一矩阵D_{11}、D_{12}、D_{21}、D_{22}，以及第二矩阵E_{11}、E_{12}、E_{21}、E_{22}。可以将第一矩阵和第二矩阵作为矩阵的一个元素执行矩阵乘法的运算过程，例如，矩阵D第一行乘以

矩阵E第一列为$F_{11}=D_{11}\times E_{11}+D_{12}\times E_{21}$，矩阵D第一行乘以矩阵E第二列为$F_{12}=D_{11}\times E_{12}+D_{12}\times E_{22}$，矩阵D第二行乘以矩阵E第一列为$F_{21}=D_{21}\times E_{11}+D_{22}\times E_{21}$，矩阵D第二行乘以矩阵E第二列为$F_{22}=D_{21}\times E_{12}+D_{22}\times E_{22}$。也就是说，为了得到最终的矩阵乘法的运算结果，需要先得到第二中间结果：

[0108] $D_{11}\times E_{11}$，$D_{12}\times E_{21}$，$D_{11}\times E_{12}$，$D_{12}\times E_{22}$，

[0109] $D_{21}\times E_{11}$，$D_{22}\times E_{21}$，$D_{21}\times E_{12}$，$D_{22}\times E_{22}$。

[0110] 具体计算第二中间结果的过程可以通过将对应的第一矩阵和第二矩阵分别按照步骤S11-S13的过程进行运算得到。

[0111] 通过对输入矩阵进行分块，并针对分块后的矩阵分别进行本公开的矩阵乘法运算得到第二中间结果，利用矩阵乘的规则根据第二中间结果可以计算得到输入矩阵的乘积。根据本公开上述实施方式的运算方法，对于任何维度的矩阵都可以快速的实现矩阵相乘的过程，运算效率高。

[0112] 对于进行分块的情况，如果处理元件包含的寄存器的数量可以满足存储输入矩阵的需求，那么还可以采用堆叠存储的方式将输入矩阵存储到处理元件的寄存器中，来实现输入矩阵的乘法运算。比如说，每个处理元件中可以包括多个寄存器，控制器可以将处理元件中的寄存器分为多组寄存器，那么，所述处理器包括多组寄存器，每组寄存器用于存储分块后的一个第一矩阵。因此，在一种可能的实现方式中，控制器可以根据对输入矩阵分块的方式对处理元件的寄存器进行分组得到多组寄存器。

[0113] 在本实施方式中，本公开的运算方法还可以包括：

[0114] 在对所述输入矩阵进行分块后，控制器在所述多组寄存器中堆叠存储所述两个以上第一矩阵，每组寄存器存储一个第一矩阵。

[0115] 在另一种可能的实现方式中，控制器也可以每次存储一个第一矩阵，参照图5的示例，根据第二中间结果计算输入矩阵的乘积。

[0116] 按照步骤S11-步骤S13的过程执行第一矩阵和与第一矩阵对应的第二矩阵的矩阵乘法运算得到第二中间结果，根据第二中间结果计算输入矩阵的乘积。其中，与第一矩阵对应的第二矩阵可以是指根据矩阵乘法规则左乘矩阵/右乘矩阵分块得到的矩阵中需要与第一矩阵进行乘法运算的矩阵。

[0117] 示例3堆叠存储结合步骤S11-步骤S13

[0118] 举例来说，以处理元件为2×2的阵列，输入矩阵都为4×4矩阵为例对本公开的运算方法进行说明。

[0119] 假设左乘矩阵$a_{44}=\begin{bmatrix}A_{11} & A_{12} & A_{13} & A_{14}\\ A_{21} & A_{22} & A_{23} & A_{24}\\ A_{31} & A_{32} & A_{33} & A_{34}\\ A_{41} & A_{42} & A_{43} & A_{44}\end{bmatrix}$，右乘矩阵为$b_{44}=\begin{bmatrix}B_{11} & B_{12} & B_{13} & B_{14}\\ B_{21} & B_{22} & B_{23} & B_{24}\\ B_{31} & B_{32} & B_{33} & B_{34}\\ B_{41} & B_{42} & B_{43} & B_{44}\end{bmatrix}$。那么，在一示例中，可以将左乘矩阵和右乘矩阵都划分为2×2的矩阵。需要说明的是，以上分块方式仅仅是本公开的一个示例，还可以采用其他方式进行分块，本公开对此不作限定。

[0120] 图6示出根据本公开一实施例的对矩阵划分的示例。如图6所示，可以将左乘矩阵和右乘矩阵都划分为2×2的子矩阵，左乘矩阵划分后得到四个第一矩阵a_{11}、a_{12}、a_{21}、a_{22}，其中，a_{11}为$\begin{bmatrix}A_{11} & A_{12}\\ A_{21} & A_{22}\end{bmatrix}$、$a_{12}$为$\begin{bmatrix}A_{13} & A_{14}\\ A_{23} & A_{24}\end{bmatrix}$、$a_{21}$为$\begin{bmatrix}A_{31} & A_{32}\\ A_{41} & A_{42}\end{bmatrix}$、$a_{22}$为$\begin{bmatrix}A_{33} & A_{34}\\ A_{43} & A_{44}\end{bmatrix}$；右乘矩阵划分后得到四个第

二矩阵b_{11}、b_{12}、b_{21}、b_{22}，其中，b_{11}为$\begin{bmatrix}B_{11} & B_{12}\\ B_{21} & B_{22}\end{bmatrix}$、$b_{12}$为$\begin{bmatrix}B_{13} & B_{14}\\ B_{23} & B_{24}\end{bmatrix}$、$b_{21}$为$\begin{bmatrix}B_{31} & B_{32}\\ B_{41} & B_{42}\end{bmatrix}$、$b_{22}$为$\begin{bmatrix}B_{33} & B_{34}\\ B_{43} & B_{44}\end{bmatrix}$。

[0121] 以采用步骤S11-S13的过程计算第二中间结果为例，假设处理元件为2×2的阵列，以图6所示的示例为例，对于本公开的运算方法，可以加载第一矩阵，加载的结果如表1所示。其中，Reg0、Reg1、Reg2和Reg3分别表示处理元件中上的一组寄存器，处理元件为2×2的阵列，每个处理元件都包括多个寄存器，在进行数据存储时用位于同一组的寄存器存储一个第一矩阵如表1所示。

[0122] 在一种可能的实现方式中，根据步骤S12的方式对第一矩阵和对应的第二矩阵进行处理：Reg0存储a_{11}、将b_{11}的第一列存储到a_{11}的第一行和第二行所在的处理元件的寄存器中，Reg1存储a_{12}、将b_{21}的第一列存储到a_{12}的第一行和第二行所在的处理元件的寄存器中，Reg2存储a_{21}、将b_{12}的第一列存储到a_{21}的第一行和第二行所在的处理元件的寄存器中，Reg3存储a_{22}、将b_{22}的第一列存储到a_{22}的第一行和第二行所在的处理元件的寄存器中，如表2所示。

[0123] 然后处理器中的控制器控制处理元件分别对对应的寄存器内存储的元素求乘积得到元素乘积，然后计算每一行元素乘积的和得到第一中间结果(具体过程，可以如上文的示例所述，不再赘述)。对于b_{11}、b_{12}、b_{21}、b_{22}的第二列，采用类似的方式进行存储并计算乘积得到元素乘积，按行求和得到第一中间结果。将第一中间结果进行处理可以得到第二中间结果$a_{11}\times b_{11}$、$a_{12}\times b_{21}$、$a_{21}\times b_{12}$以及$a_{22}\times b_{22}$。

[0124] 表1元素存储示例

[0125]

Reg0		Reg1	
A_{11}	A_{12}	A_{13}	A_{14}
A_{21}	A_{22}	A_{23}	A_{24}
Reg2		Reg3	
A_{31}	A_{32}	A_{33}	A_{34}
A_{41}	A_{42}	A_{43}	A_{44}

[0126] 表2元素存储示例

[0127]

Reg0		Reg1	
B_{11}	B_{21}	B_{31}	B_{41}
B_{11}	B_{21}	B_{31}	B_{41}
Reg2		Reg3	
B_{13}	B_{23}	B_{33}	B_{43}
B_{13}	B_{23}	B_{33}	B_{43}

[0128] 也就是说，在计算过程中，对于每一组寄存器内的元素，控制器可以控制处理元件计算得到第二中间结果$a_{11}\times b_{11}$、$a_{12}\times b_{21}$、$a_{21}\times b_{12}$以及$a_{22}\times b_{22}$。具体过程不再赘述。根据第二中间结果$a_{11}\times b_{11}$、$a_{12}\times b_{21}$、$a_{21}\times b_{12}$以及$a_{22}\times b_{22}$，控制器可以控制处理元件计算得到$C_{11}=a_{11}\times b_{11}+a_{12}\times b_{21}$，$C_{22}=a_{21}\times b_{12}+a_{22}\times b_{22}$。

[0129] 根据以上过程，控制器还可以控制处理元件根据步骤S11-S13的过程计算得到第二中间结果$a_{11}\times b_{12}$、$a_{12}\times b_{22}$、$a_{21}\times b_{11}$以及$a_{22}\times b_{21}$：将b_{11}的第一列存储到a_{21}的第一行和第二行所在的处理元件的寄存器中，将b_{21}的第一列存储到a_{22}的第一行和第二行所在的处理元件的寄存器中，将b_{12}的第一列存储到a_{11}的第一行和第二行所在的处理元件的寄存器中，将b_{22}的第一列存储到a_{12}的第一行和第二行所在的处理元件的寄存器中，然后处理器中的控制器控制处理元件分别对对应的寄存器内存储的元素求乘积得到元素乘积，然后计算每一行元素乘积的和得到第一中间结果；对b_{11}、b_{12}、b_{21}、b_{22}的第二列，采用类似的方式进行存储并计算乘积，按行求和得到第一中间结果，将第一中间结果进行处理可以得到第二中间结果$a_{11}\times b_{12}$、$a_{12}\times b_{22}$、$a_{21}\times b_{11}$以及$a_{22}\times b_{21}$。根据第二中间结果$a_{11}\times b_{12}$、$a_{12}\times b_{22}$、$a_{21}\times b_{11}$以及$a_{22}\times b_{21}$可以计算得到$C_{12}=a_{11}\times b_{12}+a_{12}\times b_{22}$，$C_{21}=a_{21}\times b_{11}+a_{22}\times b_{21}$。

[0130] 在另一种可能的实现方式中，如表3所示，在步骤S12中，控制器还可以先将b_{11}的第一列存储到a_{11}的第一行和第二行所在的处理元件的寄存器中、a_{21}的第一行和第二行所在的处理元件的寄存器中，将b_{21}的第一列存储到a_{12}的第一行和第二行所在的处理元件的寄存器中、a_{22}的第一行和第二行所在的处理元件的寄存器中。

[0131] 表3元素存储示例

[0132]

Reg0		Reg1	
B_{11}	B_{21}	B_{31}	B_{41}
B_{11}	B_{21}	B_{31}	B_{41}
Reg2		Reg3	
B_{11}	B_{21}	B_{31}	B_{41}
B_{11}	B_{21}	B_{31}	B_{41}

[0133] 对于表3的示例，处理器中的控制器控制处理元件分别对对应的寄存器内存储的元素求乘积得到元素乘积，然后计算每一行元素乘积的和得到第一中间结果。对于b_{11}、b_{21}的第二列，采用类似的方式进行存储并计算乘积得到元素乘积，按行求和得到第一中间结果。控制器可以控制处理元件根据第一中间结果计算得到第二中间结果$a_{11}\times b_{11}$、$a_{12}\times b_{21}$、$a_{21}\times b_{11}$以及$a_{22}\times b_{21}$。

[0134] 对于b_{12}、b_{22}也可以重复上述过程得到第二中间结果$a_{11}\times b_{12}$、$a_{12}\times b_{22}$、$a_{21}\times b_{12}$以及$a_{22}\times b_{22}$。具体过程不再赘述。

[0135] 根据第二中间结果可以计算得到输入矩阵的乘积。

[0136] 根据以上过程，可以采用分块的方式计算得到输入矩阵的乘积。因此，根据本公开的矩阵乘的运算方法可以实现任意大小规模的矩阵运算。并且，相比于相关技术中的矩阵乘运算可以减少访存次数，降低带宽压力，提高运算的效率。

[0137] 需要说明的是，对于前述的各方法实施例，为了简单描述，故将其都表述为一系列的动作处理，但是本领域技术人员应该知悉，本公开并不受所描述的动作顺序的限制，因为依据本公开，某些步骤可以采用其他顺序或者同时进行。其次，本领域技术人员也应该知悉，说明书中所描述的实施例均属于可选实施例，所涉及的动作和模块并不一定是本公开所必须的。

[0138] 进一步需要说明的是,虽然流程图中的各个步骤按照箭头的指示依次显示,但是这些步骤并不是必然按照箭头指示的顺序依次执行。除非本文中有明确的说明,这些步骤的执行并没有严格的顺序限制,这些步骤可以以其它的顺序执行。而且,流程图中的至少一部分步骤可以包括多个子步骤或者多个阶段,这些子步骤或者阶段并不必然是在同一时刻执行完成,而是可以在不同的时刻执行,这些子步骤或者阶段的执行顺序也不必然是依次进行,而是可以与其它步骤或者其它步骤的子步骤或者阶段的至少一部分轮流或者交替地执行。

[0139] 本公开还提供了一种处理器。图1所示为处理器的一个示例,处理器可以包括两个以上处理元件,两个以上处理元件以二维矩阵排列,每个处理元件包括至少一个寄存器,所述处理器用于实现对第一矩阵和第二矩阵的矩阵乘法运算。

[0140] 在一种可能的实现方式中,所述处理器还包括控制器,所述控制器用于将第一矩阵加载到处理元件的寄存器中;

[0141] 针对第二矩阵的每一行,所述控制器用于将所述每一行中的元素存储到第一矩阵的每一列元素存储的处理元件的寄存器,与第一矩阵的每一列中的元素分别求乘积,计算一列乘积的和得到第一中间结果;或者,针对第二矩阵的每一列,所述控制器用于将所述每一列中的元素存储到第一矩阵的每一行元素存储的处理元件的寄存器,与第一矩阵的每一行中的元素分别求乘积,计算一行乘积的和得到第一中间结果;

[0142] 所述控制器还用于将第一中间结果进行处理得到第一矩阵和第二矩阵的乘积。

[0143] 其中的第一矩阵可以是对待加载矩阵分块后得到的多个第一矩阵中的一个,待加载矩阵可以为左乘矩阵或者右乘矩阵。输入矩阵中除了待加载矩阵以外的另一个矩阵为第二矩阵。

[0144] 第一矩阵也可以不是分块后的矩阵,例如,第一矩阵可以为输入矩阵中的左乘矩阵或者右乘矩阵,第二矩阵为输入矩阵中的另一个矩阵。

[0145] 也就是说,在一种可能的实现方式中,本公开的处理器的控制器还可以根据处理元件的排列,从输入矩阵中确定不需要进行分块的矩阵为第一矩阵,输入矩阵中的另一矩阵为第二矩阵,输入矩阵包括左乘矩阵和右乘矩阵。

[0146] 在一种可能的实现方式中,第一矩阵为左乘矩阵、第二矩阵为右乘矩阵,针对第二矩阵中的每一列元素,所述控制器用于将该列元素中的每个元素存储至第一矩阵中对应的一列元素存储的处理元件的寄存器,控制每一个处理元件对相应的寄存器内的元素进行乘法运算得到元素乘积,计算每一行元素乘积的和得到第一中间结果,其中,第一矩阵中与所述每个元素对应的一列元素是指,该元素在所述第二矩阵中的行数与一列元素的列数相同。

[0147] 在另一种可能的实现方式中,第一矩阵为右乘矩阵、第二矩阵为左乘矩阵,针对第二矩阵中的每一行元素,所述控制器用于将该行元素中的每个元素存储至第一矩阵中对应的一行元素存储的处理元件的寄存器,控制每一个处理元件对相应的寄存器内的元素进行乘法运算得到元素乘积,计算每一列元素乘积的和得到第一中间结果,其中,第一矩阵中与所述每个元素对应的一行元素是指,该元素在所述第二矩阵中的列数与一行元素所在的行数相同。

[0148] 针对上述两种实施方式,对于不分块的具体的示例可以参见上文运算方法部分的

描述,不再赘述。

[0149] 在另一种可能的实现方式中,控制器还用于从输入矩阵中确定待加载矩阵;其中,输入矩阵包括左乘矩阵和右乘矩阵,待加载矩阵为左乘矩阵或右乘矩阵;根据处理元件的排列以及待加载矩阵的行秩以及列秩确定是否对待加载矩阵进行分块;若要对待加载矩阵进行分块,则所述控制器用于根据待处理元件的排列以及待加载矩阵的行秩以及列秩对待加载矩阵进行分块得到两个以上第一矩阵。

[0150] 在该实施方式中,所述控制器还用于根据对待加载矩阵分块的方式,对输入矩阵中除了待加载矩阵以外的另一个矩阵进行分块得到两个以上第二矩阵;在该实施方式中,所述处理器包括多组寄存器,在对所述输入矩阵进行分块后,所述控制器还用于在所述多组寄存器中堆叠存储所述两个以上第一矩阵,每组存储一个第一矩阵。在该实施方式中,控制器还可以根据第一矩阵和对应的第二矩阵的乘积,按照矩阵乘的规则计算所述左乘矩阵和所述右乘矩阵的乘积。

[0151] 针对上述分块的具体示例,可以参见上文中关于图5和图6部分的描述,不再赘述。

[0152] 本公开实施例还提出一种人工智能芯片,所述芯片包括如上所述的处理器。

[0153] 在一种可能的实现方式中,还公开了一种板卡,其包括存储器件、接口装置和控制器件以及上述人工智能芯片;其中,所述人工智能芯片与所述存储器件、所述控制器件以及所述接口装置分别连接;所述存储器件,用于存储数据;所述接口装置,用于实现所述人工智能芯片与外部设备之间的数据传输;所述控制器件,用于对所述人工智能芯片的状态进行监控。

[0154] 图7示出根据本公开实施例的板卡的结构框图,参阅图7,上述板卡除了包括上述芯片389以外,还可以包括其他的配套部件,该配套部件包括但不限于:存储器件390、接口装置391和控制器件392;

[0155] 所述存储器件390与所述人工智能芯片通过总线连接,用于存储数据。所述存储器件可以包括多组存储单元393。每一组所述存储单元与所述人工智能芯片通过总线连接。可以理解,每一组所述存储单元可以是DDR SDRAM(英文:Double Data Rate SDRAM,双倍速率同步动态随机存储器)。

[0156] DDR不需要提高时钟频率就能加倍提高SDRAM的速度。DDR允许在时钟脉冲的上升沿和下降沿读出数据。DDR的速度是标准SDRAM的两倍。在一个实施例中,所述存储装置可以包括4组所述存储单元。每一组所述存储单元可以包括多个DDR4颗粒(芯片)。在一个实施例中,所述人工智能芯片内部可以包括4个72位DDR4控制器,上述72位DDR4控制器中64bit用于传输数据,8bit用于ECC校验。

[0157] 在一个实施例中,每一组所述存储单元包括多个并联设置的双倍速率同步动态随机存储器。DDR在一个时钟周期内可以传输两次数据。在所述芯片中设置控制DDR的控制器,用于对每个所述存储单元的数据传输与数据存储的控制。

[0158] 所述接口装置与所述人工智能芯片电连接。所述接口装置用于实现所述人工智能芯片与外部设备(例如服务器或计算机)之间的数据传输。例如在一个实施例中,所述接口装置可以为标准PCIE接口。比如,待处理的数据由服务器通过标准PCIE接口传递至所述芯片,实现数据转移。在另一个实施例中,所述接口装置还可以是其他的接口,本公开并不限制上述其他的接口的具体表现形式,所述接口单元能够实现转接功能即可。另外,所述人工

智能芯片的计算结果仍由所述接口装置传送回外部设备(例如服务器)。

[0159] 所述控制器件与所述人工智能芯片电连接。所述控制器件用于对所述人工智能芯片的状态进行监控。具体的,所述人工智能芯片与所述控制器件可以通过SPI接口电连接。所述控制器件可以包括单片机(Micro Controller Unit,MCU)。如所述人工智能芯片可以包括多个处理芯片、多个处理核或多个处理电路,可以带动多个负载。因此,所述人工智能芯片可以处于多负载和轻负载等不同的工作状态。通过所述控制装置可以实现对所述人工智能芯片中多个处理芯片、多个处理和/或多个处理电路的工作状态的调控。

[0160] 本公开实施例还提出一种计算机可读存储介质,其上存储有计算机程序指令,所述计算机程序指令被处理器执行时实现上述方法。计算机可读存储介质可以是非易失性计算机可读存储介质。

[0161] 本公开实施例还提出一种电子设备,包括上述处理器。

[0162] 应该理解,上述的实施例仅是示意性的,本公开的装置还可通过其它的方式实现。例如,上述实施例中所述单元/模块的划分,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。例如,多个单元、模块或组件可以结合,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。

[0163] 另外,若无特别说明,在本公开各个实施例中的各功能单元/模块可以集成在一个单元/模块中,也可以是各个单元/模块单独物理存在,也可以两个或两个以上单元/模块集成在一起。上述集成的单元/模块既可以采用硬件的形式实现,也可以采用软件程序模块的形式实现。

[0164] 所述集成的单元/模块如果以硬件的形式实现时,该硬件可以是数字电路,模拟电路等等。硬件结构的物理实现包括但不局限于晶体管,忆阻器等等。

[0165] 所述集成的单元/模块如果以软件程序模块的形式实现并作为独立的产品销售或使用时,可以存储在一个计算机可读取存储器中。基于这样的理解,本公开的技术方案本质上或者说对现有技术做出贡献的部分或者该技术方案的全部或部分可以以软件产品的形式体现出来,该计算机软件产品存储在一个存储器中,包括若干指令用以使得一台计算机设备(可为个人计算机、服务器或者网络设备等)执行本公开各个实施例所述方法的全部或部分步骤。而前述的存储器包括:U盘、只读存储器(ROM,Read-Only Memory)、随机存取存储器(RAM,Random Access Memory)、移动硬盘、磁碟或者光盘等各种可以存储程序代码的介质。

[0166] 在上述实施例中,对各个实施例的描述都各有侧重,某个实施例中没有详述的部分,可以参见其他实施例的相关描述。上述实施例的各技术特征可以进行任意的处理,为使描述简洁,未对上述实施例中的各个技术特征所有可能的处理都进行描述,然而,只要这些技术特征的处理不存在矛盾,都应当认为是本说明书记载的范围。

[0167] 本公开可以是系统、方法和/或计算机程序产品。计算机程序产品可以包括计算机可读存储介质,其上载有用于使处理器实现本公开的各个方面的计算机可读程序指令。

[0168] 计算机可读存储介质可以是可以保持和存储由指令执行设备使用的指令的有形设备。计算机可读存储介质例如可以是——但不限于——电存储设备、磁存储设备、光存储设备、电磁存储设备、半导体存储设备或者上述的任意合适的处理。计算机可读存储介质的更具体的例子(非穷举的列表)包括:便携式计算机盘、硬盘、随机存取存储器(RAM)、只读存

储器(ROM)、可擦式可编程只读存储器(EPROM或闪存)、静态随机存取存储器(SRAM)、便携式压缩盘只读存储器(CD-ROM)、数字多功能盘(DVD)、记忆棒、软盘、机械编码设备、例如其上存储有指令的打孔卡或凹槽内凸起结构、以及上述的任意合适的处理。这里所使用的计算机可读存储介质不被解释为瞬时信号本身,诸如无线电波或者其他自由传播的电磁波、通过波导或其他传输媒介传播的电磁波(例如,通过光纤电缆的光脉冲)、或者通过电线传输的电信号。

[0169] 这里所描述的计算机可读程序指令可以从计算机可读存储介质下载到各个计算/处理设备,或者通过网络、例如因特网、局域网、广域网和/或无线网下载到外部计算机或外部存储设备。网络可以包括铜传输电缆、光纤传输、无线传输、路由器、防火墙、交换机、网关计算机和/或边缘服务器。每个计算/处理设备中的网络适配卡或者网络接口从网络接收计算机可读程序指令,并转发该计算机可读程序指令,以供存储在各个计算/处理设备中的计算机可读存储介质中。

[0170] 用于执行本公开操作的计算机程序指令可以是汇编指令、指令集架构(ISA)指令、机器指令、机器相关指令、微代码、固件指令、状态设置数据、或者以一种或多种编程语言的任意处理编写的源代码或目标代码,所述编程语言包括面向对象的编程语言—诸如Smalltalk、C++等,以及常规的过程式编程语言—诸如“C”语言或类似的编程语言。计算机可读程序指令可以完全地在用户计算机上执行、部分地在用户计算机上执行、作为一个独立的软件包执行、部分在用户计算机上部分在远程计算机上执行、或者完全在远程计算机或服务器上执行。在涉及远程计算机的情形中,远程计算机可以通过任意种类的网络—包括局域网(LAN)或广域网(WAN)—连接到用户计算机,或者,可以连接到外部计算机(例如利用因特网服务提供商来通过因特网连接)。在一些实施例中,通过利用计算机可读程序指令的状态信息来个性化定制电子电路,例如可编程逻辑电路、现场可编程门阵列(FPGA)或可编程逻辑阵列(PLA),该电子电路可以执行计算机可读程序指令,从而实现本公开的各个方面。

[0171] 这里参照根据本公开实施例的方法、装置(系统)和计算机程序产品的流程图和/或框图描述了本公开的各个方面。应当理解,流程图和/或框图的每个方框以及流程图和/或框图中各方框的处理,都可以由计算机可读程序指令实现。

[0172] 这些计算机可读程序指令可以提供给通用计算机、专用计算机或其它可编程数据处理装置的处理器,从而生产出一种机器,使得这些指令在通过计算机或其它可编程数据处理装置的处理器执行时,产生了实现流程图和/或框图中的一个或多个方框中规定的功能/动作的装置。也可以把这些计算机可读程序指令存储在计算机可读存储介质中,这些指令使得计算机、可编程数据处理装置和/或其他设备以特定方式工作,从而,存储有指令的计算机可读介质则包括一个制造品,其包括实现流程图和/或框图中的一个或多个方框中规定的功能/动作的各个方面的指令。

[0173] 也可以把计算机可读程序指令加载到计算机、其它可编程数据处理装置、或其它设备上,使得在计算机、其它可编程数据处理装置或其它设备上执行一系列操作步骤,以产生计算机实现的过程,从而使得在计算机、其它可编程数据处理装置、或其它设备上执行的指令实现流程图和/或框图中的一个或多个方框中规定的功能/动作。

[0174] 附图中的流程图和框图显示了根据本公开的多个实施例的系统、方法和计算机程

序产品的可能实现的体系架构、功能和操作。在这点上,流程图或框图中的每个方框可以代表一个模块、程序段或指令的一部分,所述模块、程序段或指令的一部分包含一个或多个用于实现规定的逻辑功能的可执行指令。在有些作为替换的实现中,方框中所标注的功能也可以以不同于附图中所标注的顺序发生。例如,两个连续的方框实际上可以基本并行地执行,它们有时也可以按相反的顺序执行,这依所涉及的功能而定。也要注意的是,框图和/或流程图中的每个方框、以及框图和/或流程图中的方框的处理,可以用执行规定的功能或动作的专用的基于硬件的系统来实现,或者可以用专用硬件与计算机指令的处理来实现。

[0175] 依据以下条款可更好地理解前述内容:

[0176] 条款A1.一种基于处理元件矩阵的矩阵乘的运算方法,应用于处理器,所述处理器包括两个以上处理元件,所述两个以上处理元件以二维矩阵排列,处理元件包括至少一个寄存器,所述方法实现对第一矩阵和第二矩阵的矩阵乘法运算,

[0177] 所述方法包括:

[0178] 将第一矩阵加载到处理元件的寄存器中;

[0179] 针对第二矩阵的每一行,将所述每一行中的元素存储到第一矩阵的每一列元素存储的处理元件的寄存器,与第一矩阵的每一列中的元素分别求乘积,计算一列乘积的和得到第一中间结果;或者,针对第二矩阵的每一列,将所述每一列中的元素存储到第一矩阵的每一行元素存储的处理元件的寄存器,与第一矩阵的每一行中的元素分别求乘积,计算一行乘积的和得到第一中间结果;

[0180] 将第一中间结果进行处理得到第一矩阵和第二矩阵的乘积。

[0181] 条款A2.根据条款A1所述的方法,第一矩阵为左乘矩阵、第二矩阵为右乘矩阵,

[0182] 针对第二矩阵中的每一列元素,将该列元素中的每个元素存储至第一矩阵中对应的一列元素存储的处理元件的寄存器,控制每一个处理元件对相应的寄存器内的元素进行乘法运算得到元素乘积,计算每一行元素乘积的和得到第一中间结果,

[0183] 其中,第一矩阵中与所述每个元素对应的一列元素是指,该元素在所述第二矩阵中的行数与一列元素的列数相同。

[0184] 条款A3.根据条款A1所述的方法,第一矩阵为右乘矩阵、第二矩阵为左乘矩阵,

[0185] 针对第二矩阵中的每一行元素,将该行元素中的每个元素存储至第一矩阵中对应的一行元素存储的处理元件的寄存器,控制每一个处理元件对相应的寄存器内的元素进行乘法运算得到元素乘积,计算每一列元素乘积的和得到第一中间结果,

[0186] 其中,第一矩阵中与所述每个元素对应的一行元素是指,该元素在所述第二矩阵中的列数与一行元素所在的行数相同。

[0187] 条款A4.根据条款A1-A3任意一项所述的方法,所述方法还包括:

[0188] 根据处理元件的排列,从输入矩阵中确定不需要进行分块的矩阵为第一矩阵,输入矩阵中的另一矩阵为第二矩阵。

[0189] 条款A5.根据条款A1-A3任意一项所述的方法,所述方法还包括:

[0190] 从输入矩阵中确定待加载矩阵;其中,输入矩阵包括左乘矩阵和右乘矩阵,待加载矩阵为左乘矩阵或右乘矩阵;

[0191] 根据处理元件的排列以及待加载矩阵的行秩以及列秩确定是否对待加载矩阵进行分块;其中,待加载矩阵为左乘矩阵或右乘矩阵;

[0192] 若要对待加载矩阵进行分块,则根据待处理元件的排列以及待加载矩阵的行秩以及列秩对待加载矩阵进行分块得到两个以上第一矩阵。

[0193] 条款A6.根据条款A5所述的方法,所述方法还包括:

[0194] 根据对待加载矩阵分块的方式,对输入矩阵中除了待加载矩阵以外的另一个矩阵进行分块得到两个以上第二矩阵;

[0195] 根据第一矩阵和对应的第二矩阵的乘积,按照矩阵乘的规则计算所述左乘矩阵和所述右乘矩阵的乘积。

[0196] 条款A7.根据条款A5所述的方法,所述处理器包括多组寄存器,所述方法还包括:

[0197] 在对所述输入矩阵进行分块后,在所述多组寄存器中堆叠存储所述两个以上第一矩阵,每组存储一个第一矩阵。

[0198] 条款A8.一种处理器,所述处理器包括两个以上处理元件,所述两个以上处理元件以二维矩阵排列,处理元件包括至少一个寄存器,所述处理器用于对第一矩阵和第二矩阵执行矩阵乘法运算,

[0199] 所述处理器还包括控制器,所述控制器用于将第一矩阵加载到处理元件的寄存器中;

[0200] 针对第二矩阵的每一行,所述控制器用于将所述每一行中的元素存储到第一矩阵的每一列元素存储的处理元件的寄存器,与第一矩阵的每一列中的元素分别求乘积,计算一列乘积的和得到第一中间结果;或者,针对第二矩阵的每一列,所述控制器用于将所述每一列中的元素存储到第一矩阵的每一行元素存储的处理元件的寄存器,与第一矩阵的每一行中的元素分别求乘积,计算一行乘积的和得到第一中间结果;

[0201] 所述控制器还用于将第一中间结果进行处理得到第一矩阵和第二矩阵的乘积。

[0202] 条款A9.根据条款A8所述的处理器,第一矩阵为左乘矩阵、第二矩阵为右乘矩阵,

[0203] 针对第二矩阵中的每一列元素,所述控制器用于将该列元素中的每个元素存储至第一矩阵中对应的一列元素存储的处理元件的寄存器,控制每一个处理元件对相应的寄存器内的元素进行乘法运算得到元素乘积,计算每一行元素乘积的和得到第一中间结果,

[0204] 其中,第一矩阵中与所述每个元素对应的一列元素是指,该元素在所述第二矩阵中的行数与一列元素的列数相同。

[0205] 条款A10.根据条款A8所述的处理器,第一矩阵为右乘矩阵、第二矩阵为左乘矩阵,

[0206] 针对第二矩阵中的每一行元素,所述控制器用于将该行元素中的每个元素存储至第一矩阵中对应的一行元素存储的处理元件的寄存器,控制每一个处理元件对相应的寄存器内的元素进行乘法运算得到元素乘积,计算每一列元素乘积的和得到第一中间结果,

[0207] 其中,第一矩阵中与所述每个元素对应的一行元素是指,该元素在所述第二矩阵中的列数与一行元素所在的行数相同。

[0208] 条款A11.根据条款A8-A10任意一项所述的处理器,所述处理器还用于根据处理元件的排列,从输入矩阵中确定不需要进行分块的矩阵为第一矩阵,输入矩阵中的另一矩阵为第二矩阵,输入矩阵包括左乘矩阵和右乘矩阵。

[0209] 条款A12.根据条款A8-A10任意一项所述的处理器,所述控制器还用于从输入矩阵中确定待加载矩阵;其中,输入矩阵包括左乘矩阵和右乘矩阵,待加载矩阵为左乘矩阵或右乘矩阵;根据处理元件的排列以及待加载矩阵的行秩以及列秩确定是否对待加载矩阵进行

分块;

[0210] 若要对待加载矩阵进行分块,则所述控制器用于根据待处理元件的排列以及待加载矩阵的行秩以及列秩对待加载矩阵进行分块得到两个以上第一矩阵。

[0211] 条款A13.根据条款A12所述的处理器,所述控制器还用于根据对待加载矩阵分块的方式,对输入矩阵中除了待加载矩阵以外的另一个矩阵进行分块得到两个以上第二矩阵;根据第一矩阵和对应的第二矩阵的乘积,按照矩阵乘的规则计算所述左乘矩阵和所述右乘矩阵的乘积。

[0212] 条款A14.根据条款A12所述的处理器,所述处理器包括多组寄存器,在对所述输入矩阵进行分块后,所述控制器还用于在所述多组寄存器中堆叠存储所述两个以上第一矩阵,每组存储一个第一矩阵。

[0213] 条款A15.一种人工智能芯片,所述芯片包括如条款A8-A14中任意一项所述的处理器。

[0214] 条款A16.一种电子设备,包括如条款A15所述的人工智能芯片。

[0215] 以上对本公开实施例进行了详细介绍,本文中应用了具体个例对本公开的原理及实施方式进行了阐述,以上实施例的说明仅用于帮助理解本公开的方法及其核心思想。同时,本领域技术人员依据本公开的思想,基于本公开的具体实施方式及应用范围上做出的改变或变形之处,都属于本公开保护的范围。综上所述,本说明书内容不应理解为对本公开的限制。

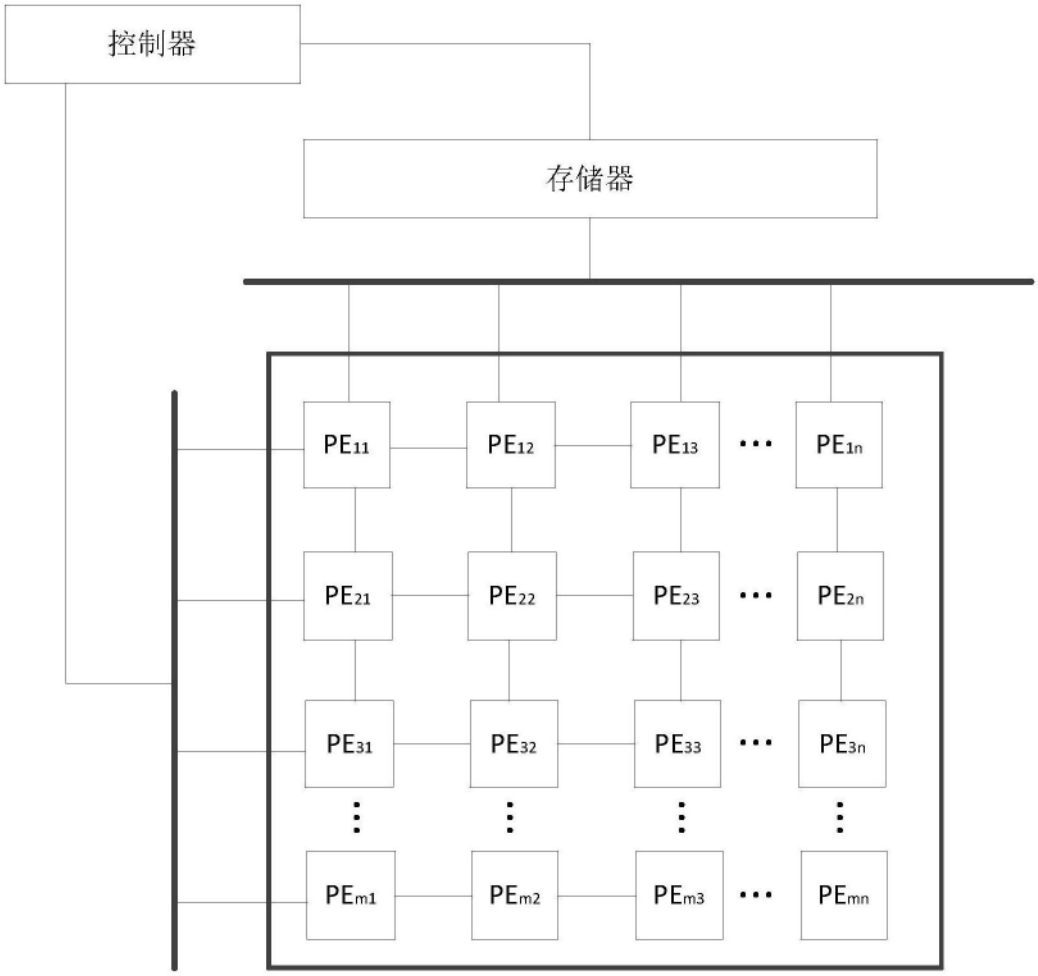

控制器
存储器
PE11
PE12
PE13
PE1n
PE21
PE22
PE23
PE2n
PE31
PE32
PE33
PE3n
PEm1
PEm2
PEm3
PEmn

图1

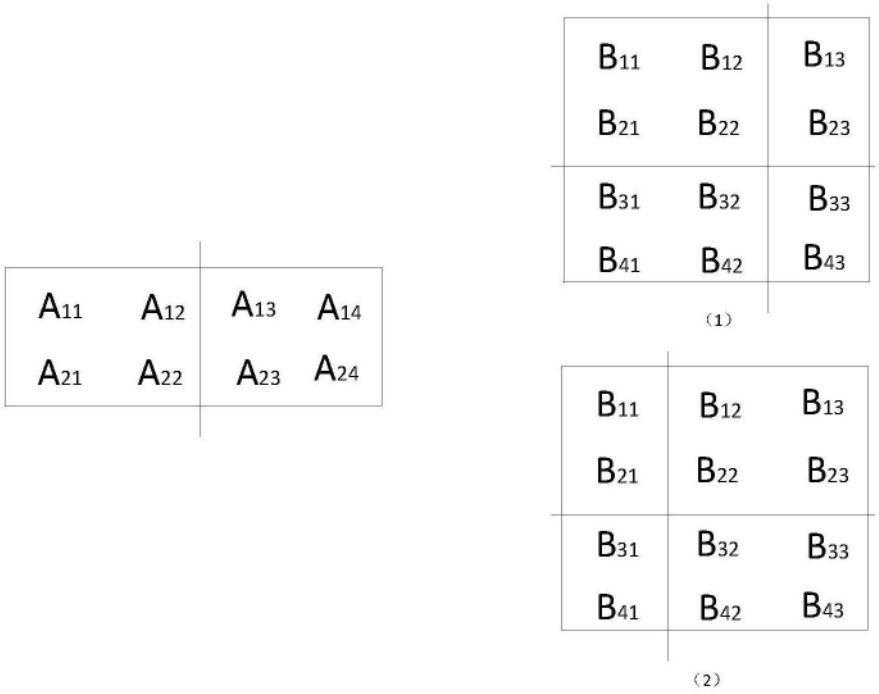

A11 A12 A13 A14
A21 A22 A23 A24
B11 B12 B13
B21 B22 B23
B31 B32 B33
B41 B42 B43
(1)
B11 B12 B13
B21 B22 B23
B31 B32 B33
B41 B42 B43
(2)

图2a

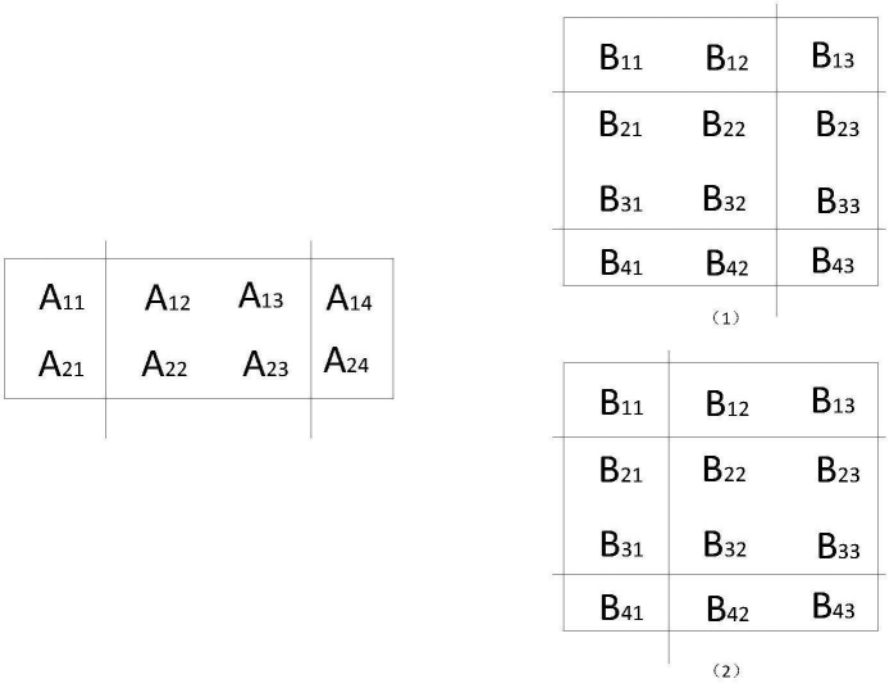

A11 A12 A13 A14
A21 A22 A23 A24
B11 B12 B13
B21 B22 B23
B31 B32 B33
B41 B42 B43
(1)
B11 B12 B13
B21 B22 B23
B31 B32 B33
B41 B42 B43
(2)

图2b

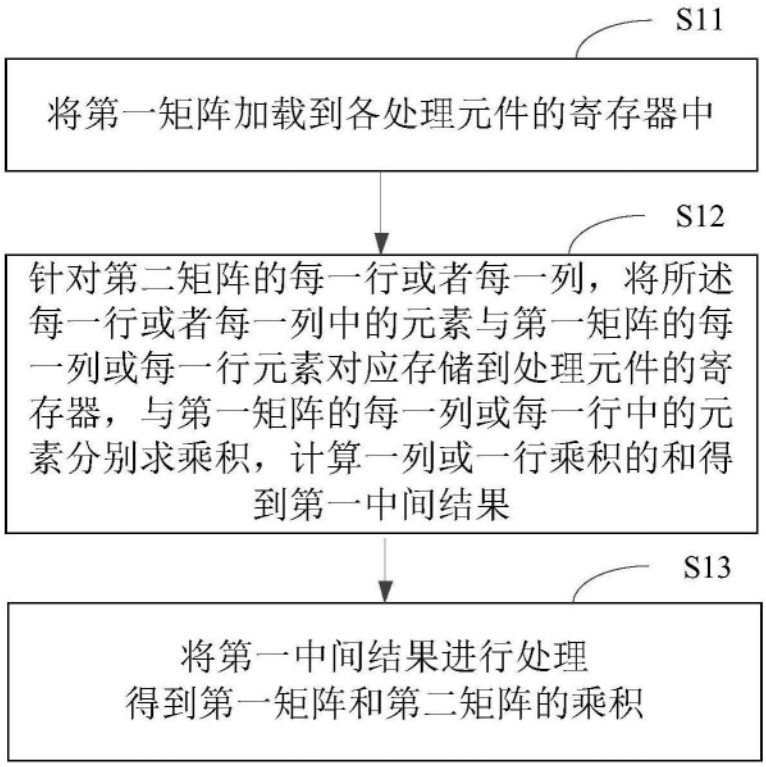

S11
将第一矩阵加载到各处理元件的寄存器中
S12
针对第二矩阵的每一行或者每一列，将所述每一行或者每一列中的元素与第一矩阵的每一列或每一行元素对应存储到处理元件的寄存器，与第一矩阵的每一列或每一行中的元素分别求乘积，计算一列或一行乘积的和得到第一中间结果
S13
将第一中间结果进行处理
得到第一矩阵和第二矩阵的乘积

图3

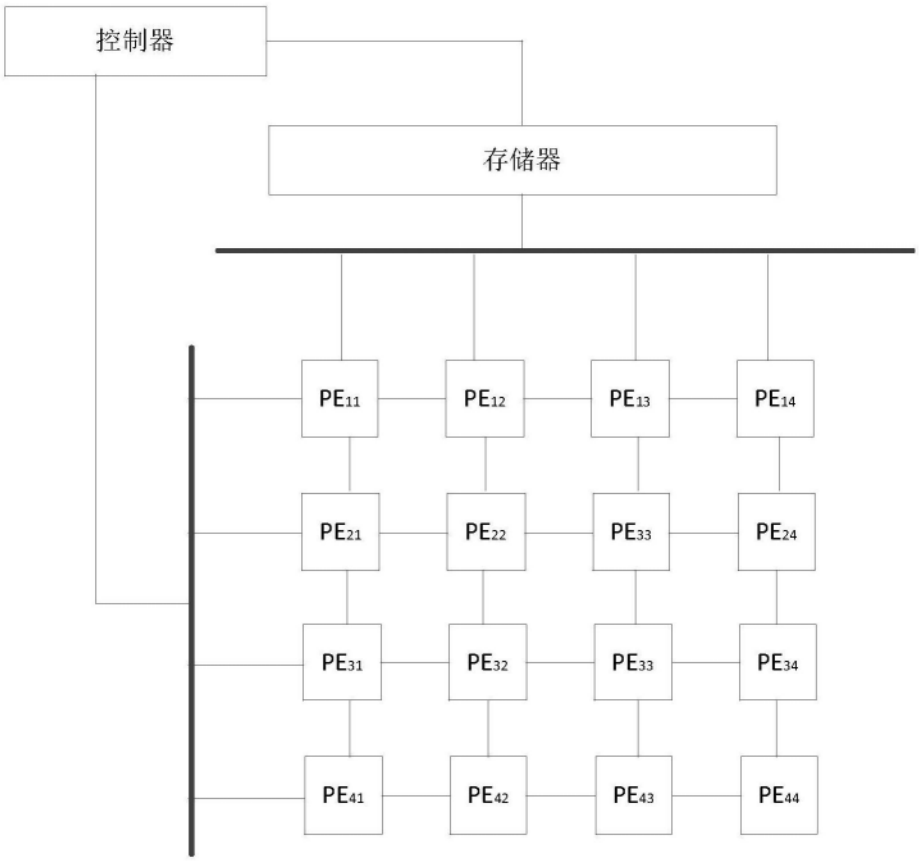

控制器
存储器
PE11
PE12
PE13
PE14
PE21
PE22
PE33
PE24
PE31
PE32
PE33
PE34
PE41
PE42
PE43
PE44

图4

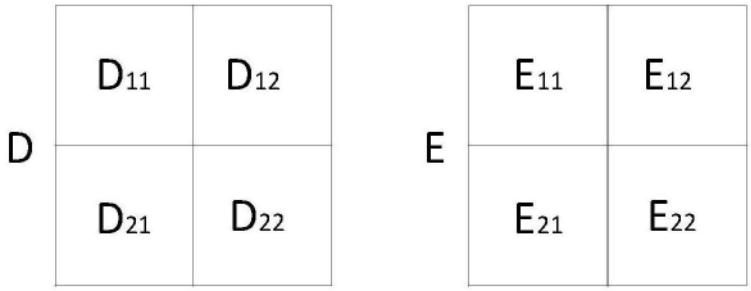

D
D11
D12
D21
D22
E
E11
E12
E21
E22

图5

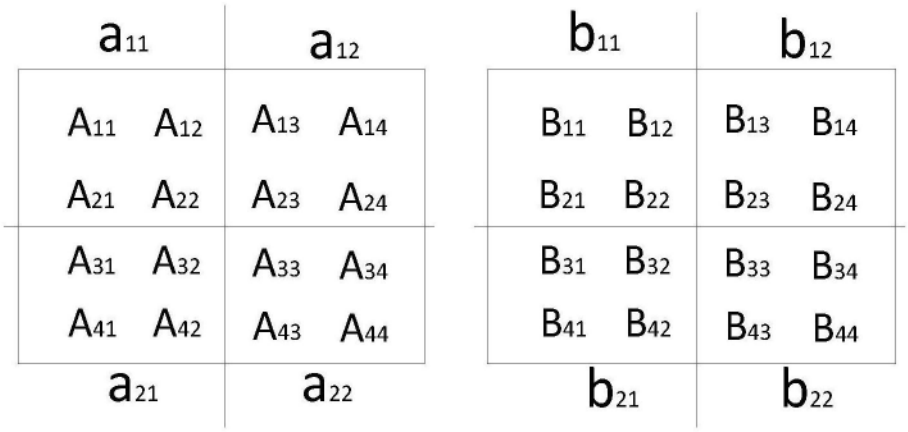

a11
a12
A11 A12 A13 A14
A21 A22 A23 A24
A31 A32 A33 A34
A41 A42 A43 A44
a21
a22
b11
b12
B11 B12 B13 B14
B21 B22 B23 B24
B31 B32 B33 B34
B41 B42 B43 B44
b21
b22

图6

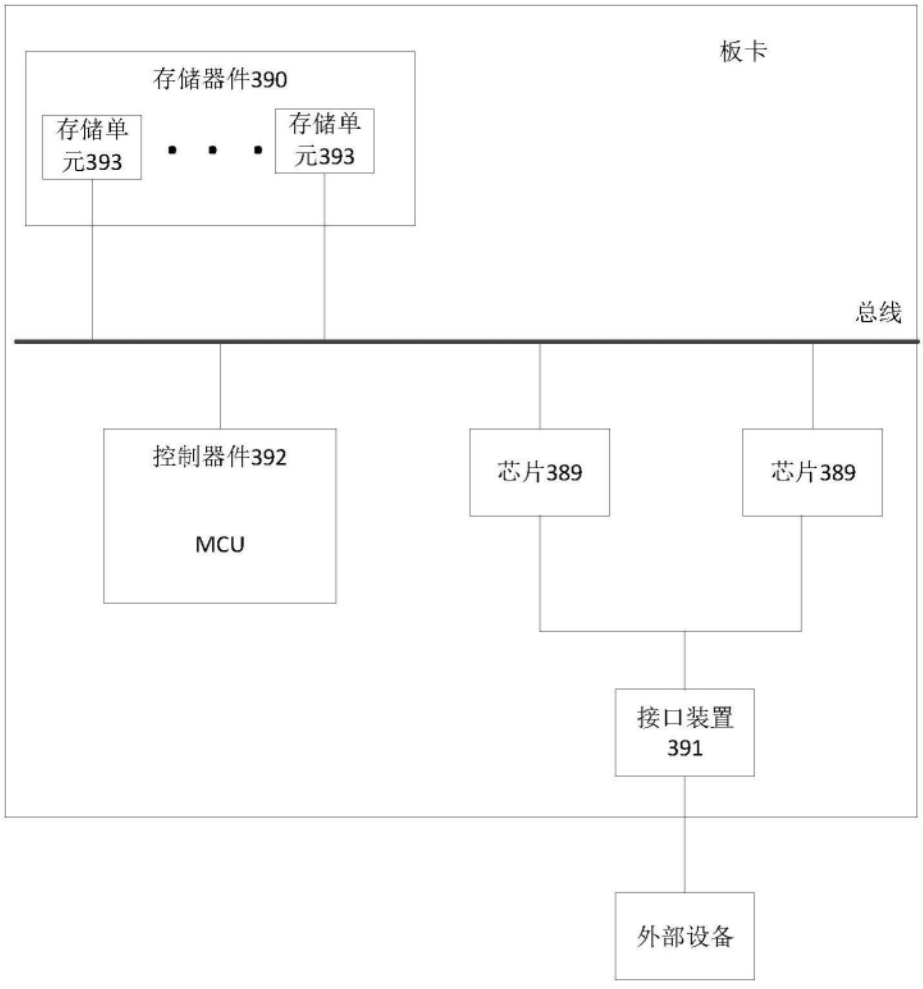

板卡
存储器件390
存储单元393
存储单元393
总线
控制器件392
MCU
芯片389
芯片389
接口装置391
外部设备

图7