(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 112506567 B
(45)授权公告日 2022.11.04

(21)申请号 202011353109.5

(22)申请日 2020.11.27

(65)同一申请的已公布的文献号
申请公布号 CN 112506567 A

(43)申请公布日 2021.03.16

(73)专利权人 海光信息技术股份有限公司
地址 300392 天津市华苑产业区海泰西路18号北2-204工业孵化-3-8

(72)发明人 左航 韩洁 卢一帆

(74)专利代理机构 北京市柳沈律师事务所 11105
专利代理师 彭久云 罗莎

(51)Int.Cl.
G06F 9/30(2006.01)
G06F 9/38(2006.01)
G06F 17/16(2006.01)

(56)对比文件
JP 2020173717 A,2020.10.22
US 2012036395 A1,2012.02.09
CN 110704019 A,2020.01.17
jakub kurzak.optimizing matrix multiplication for a shortvector simd architecture-cell processor.《elsevier》.2009,138-150.
刘伟浩.嵌入式设备可信运行环境机器学习服务的研究与实现.《中国优秀硕士学位论文全文数据库 (信息科技辑)》.2020,1-92.

审查员 李慧

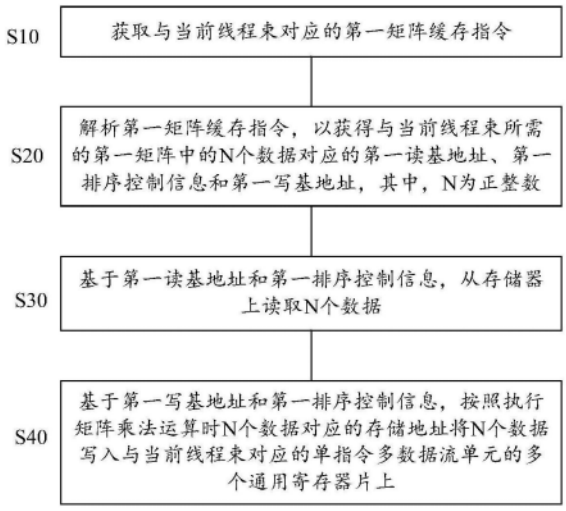
权利要求书6页 说明书31页 附图19页

(54)发明名称

数据读取方法和数据读取电路

(57)摘要

一种数据读取方法和数据读取电路。该数据读取方法应用于矩阵乘法运算,且包括:获取与当前线程束对应的第一矩阵缓存指令,其中,第一矩阵缓存指令用于实现对第一矩阵的数据进行读取;解析第一矩阵缓存指令,以获得与当前线程束所需的第一矩阵中的P个数据对应的第一读基地址、第一排序控制信息和第一写基地址,其中,P为正整数;基于第一读基地址和第一排序控制信息,从存储器上读取P个数据;基于第一写基地址和第一排序控制信息,按照执行矩阵乘法运算时P个数据对应的存储地址将P个数据写入与当前线程束对应的单指令多数据流单元的多个矢量通用寄存器片上。



CN 112506567 B

1.一种数据读取方法,应用于矩阵乘法运算,其中,所述数据读取方法包括:

获取与当前线程束对应的第一矩阵缓存指令,其中,所述第一矩阵缓存指令用于实现对第一矩阵的数据进行读取;

解析所述第一矩阵缓存指令,以获得与所述当前线程束所需的所述第一矩阵中的P个数据对应的第一读基地址、第一排序控制信息和第一写基地址,其中,P为正整数;

基于所述第一读基地址和所述第一排序控制信息,从存储器上读取所述P个数据;

基于所述第一写基地址和所述第一排序控制信息,按照执行所述矩阵乘法运算时所述P个数据对应的存储地址将所述P个数据写入与所述当前线程束对应的单指令多数据流单元的多个矢量通用寄存器片上;

其中,所述第一读基地址表示所述第一矩阵中所述P个数据所在的数据块的第一行第一列的数据或所述第一矩阵的第一行第一列的数据在所述存储器中的地址,所述第一写基地址表示读取的所述P个数据被写入所述多个矢量通用寄存器时对应的第一个索引地址。

2.根据权利要求1所述的数据读取方法,其中,解析所述第一矩阵缓存指令,以获得所述第一排序控制信息,包括:

解析所述第一矩阵缓存指令,以得到封装在所述第一矩阵缓存指令中的指令操作码、工作组尺寸、地址偏移量、地址步长、矩阵信息和运算参数,

其中,所述第一排序控制信息包括所述指令操作码、所述工作组尺寸、所述地址偏移量、所述地址步长、所述矩阵信息和所述运算参数,

所述矩阵乘法运算用于实现将第一运算矩阵和第二运算矩阵进行乘法计算,所述第一运算矩阵表示为M*K,所述第二运算矩阵表示为K*N,所述运算参数为K,其中,M、K、N均为正整数,

所述矩阵信息用于指示所述第一矩阵为所述第一运算矩阵或所述第二运算矩阵以及所述第一矩阵是否被转置,

在所述第一矩阵为所述第一运算矩阵的情况下,所述地址步长表示所述第一矩阵的相邻两列数据之间的步长,

在所述第一矩阵为所述第二运算矩阵的情况下,所述地址步长表示所述第一矩阵的相邻两行数据之间的步长。

3.根据权利要求2所述的数据读取方法,其中,基于所述第一读基地址和所述第一排序控制信息,从存储器上读取所述P个数据,包括:

根据所述第一读基地址和所述第一排序控制信息,计算得到与所述第一读基地址对应的多个存储器地址;

基于所述多个存储器地址,从所述存储器上读取多个待处理数据,其中,所述多个待处理数据表示存储在所述存储器的所述多个存储器地址中的数据;

基于所述第一排序控制信息,从所述多个待处理数据中获取所述当前线程束所需的所述P个数据。

4.根据权利要求3所述的数据读取方法,其中,所述第一矩阵包括多个第一数据块,

根据所述第一读基地址和所述第一排序控制信息,计算得到与所述第一读基地址对应的多个存储器地址,包括:

根据所述指令操作码、所述工作组尺寸和所述矩阵信息,确定所述多个第一数据块中

与所述当前线程束对应的至少一个第一数据块，其中，所述至少一个第一数据块包括所述多个待处理数据；

根据所述至少一个第一数据块的数量和所述运算参数，控制循环状态并输出循环参数；

基于所述至少一个第一数据块中的每个第一数据块的尺寸、所述循环参数、所述第一读基地址、所述地址步长和所述地址偏移量，计算得到所述多个存储器地址。

5.根据权利要求4所述的数据读取方法，其中，基于所述至少一个第一数据块中的每个第一数据块的尺寸、所述循环参数、所述第一读基地址、所述地址步长和所述地址偏移量，计算得到所述多个存储器地址，包括：

基于所述至少一个第一数据块中的每个第一数据块的尺寸、所述循环参数、所述第一读基地址、所述地址步长和所述地址偏移量，计算得到多个首地址，其中，所述至少一个第一数据块中的每个第一数据块的数据排列为多行多列，在所述第一矩阵为所述第一运算矩阵的情况下，所述多个首地址为所述至少一个第一数据块的各列数据中的第一个数据的地址，在所述第一矩阵为所述第二运算矩阵的情况下，所述多个首地址为所述至少一个第一数据块的各行数据的第一个数据的地址；

基于所述多个首地址，计算所述多个存储器地址。

6.根据权利要求2所述的数据读取方法，其中，基于所述第一写基地址和所述第一排序控制信息，按照执行所述矩阵乘法运算时所述P个数据对应的存储地址将所述P个数据写入与所述当前线程束对应的单指令多数据流单元的多个矢量通用寄存器片上，包括：

根据所述第一写基地址和所述第一排序控制信息，计算得到与所述P个数据对应的多个寄存器地址，其中，所述多个寄存器地址为所述多个矢量通用寄存器片上用于写入所述P个数据的地址，所述多个寄存器地址表示执行所述矩阵乘法运算时所述P个数据对应的存储地址；

基于所述第一排序控制信息，确定所述P个数据与所述多个寄存器地址的对应关系，基于所述对应关系，将所述P个数据写入所述多个寄存器地址中。

7.根据权利要求6所述的数据读取方法，其中，根据所述第一写基地址和所述第一排序控制信息，计算得到与所述P个数据对应的多个寄存器地址，包括：

根据所述第一写基地址、所述指令操作码、所述工作组尺寸和所述运算参数，计算得到所述多个寄存器地址。

8.根据权利要求6所述的数据读取方法，其中，基于所述对应关系，将所述P个数据写入所述多个寄存器地址中，包括：

响应于所述第一矩阵为所述第一运算矩阵：

基于所述对应关系，直接将所述P个数据写入所述多个寄存器地址中；

响应于所述第一矩阵为所述第二运算矩阵：

获取所述P个数据中的第a个数据，其中，a为正整数；

将所述第a个数据复制D份，其中，D为大于1的正整数；

基于所述对应关系，确定所述多个寄存器地址中与所述第a个数据对应的寄存器地址；

将D份所述第a个数据写入与所述第a个数据对应的寄存器地址中。

9.根据权利要求8所述的数据读取方法，其中，响应于所述第一矩阵为所述第一运算矩

阵，基于所述对应关系，直接将所述P个数据写入所述多个寄存器地址中，包括：

将所述P个数据被划分为与所述多个矢量通用寄存器片一一对应的多个数据组；

直接将所述多个数据组分别写入所述多个矢量通用寄存器片上。

10.根据权利要求1所述的数据读取方法，其中，基于所述第一读基地址和所述第一排序控制信息，从存储器上读取所述P个数据，包括：

根据所述第一读基地址和所述第一排序控制信息，计算得到与所述第一读基地址对应的多个存储器地址；

基于所述多个存储器地址，从所述存储器上读取所述P个数据，其中，所述P个数据表示存储在所述存储器的所述多个存储器地址中的数据。

11.根据权利要求1～10任一项所述的数据读取方法，其中，获取与当前线程束对应的第一矩阵缓存指令，包括：

接收与所述当前线程束对应的命令；

基于所述命令，读取所述第一矩阵缓存指令。

12.根据权利要求1～10任一项所述的数据读取方法，其中，解析所述第一矩阵缓存指令，以获得所述第一读基地址，包括：

解析所述第一矩阵缓存指令，以得到所述当前线程束所在的工作组对应的工作组基地址；

获取所述当前线程束对应的基地址偏移量；

基于所述工作组基地址和所述基地址偏移量，得到所述第一读基地址。

13.根据权利要求1～10任一项所述的数据读取方法，其中，解析所述第一矩阵缓存指令，以获得所述第一读基地址，包括：

解析所述第一矩阵缓存指令，以得到所述当前线程束所在的工作组对应的工作组基地址；

将所述工作组基地址作为所述第一读基地址。

14.根据权利要求1～10任一项所述的数据读取方法，还包括：

获取与所述当前线程束对应的第二矩阵缓存指令，其中，所述第二矩阵缓存指令用于实现对第二矩阵的数据进行读取；

解析所述第二矩阵缓存指令，以获得与所述当前线程束所需的所述第二矩阵中的Q个数据对应的第二读基地址、第二排序控制信息和第二写基地址，其中，Q为正整数；

基于所述第二读基地址和所述第二排序控制信息，从所述存储器上读取所述Q个数据；

基于所述第二写基地址和所述第二排序控制信息，按照执行所述矩阵乘法运算时所述Q个数据对应的存储地址将所述Q个数据写入所述单指令多数据流单元的多个矢量通用寄存器片上，

其中，所述当前线程束使用写入所述单指令多数据流单元的多个矢量通用寄存器片上的所述P个数据和所述Q个数据进行所述矩阵乘法运算。

15.一种数据读取电路，应用于矩阵乘法运算，包括：硬件调度电路和读取与排序电路，其中，所述硬件调度电路和所述读取与排序电路耦接，

所述硬件调度电路被配置为：

获取与当前线程束对应的第一矩阵缓存指令，其中，所述第一矩阵缓存指令用于实现

对第一矩阵的数据进行读取；

解析所述第一矩阵缓存指令，以获得与所述当前线程束所需的所述第一矩阵中的P个数据对应的第一读基地址、第一排序控制信息和第一写基地址，其中，P为正整数；

所述读取与排序电路被配置为：

基于所述第一读基地址和所述第一排序控制信息，从存储器上读取所述P个数据；

基于所述第一写基地址和所述第一排序控制信息，按照执行所述矩阵乘法运算时所述P个数据对应的存储地址将所述P个数据写入与所述当前线程束对应的单指令多数据流单元的多个矢量通用寄存器片上；

其中，所述第一读基地址表示所述第一矩阵中所述P个数据所在的数据块的第一行第一列的数据或所述第一矩阵的第一行第一列的数据在所述存储器中的地址，所述第一写基地址表示读取的所述P个数据被写入所述多个矢量通用寄存器时对应的第一个索引地址。

16.根据权利要求15所述的数据读取电路，其中，在执行解析所述第一矩阵缓存指令，以获得所述第一排序控制信息的步骤时，所述硬件调度电路被配置为：

解析所述第一矩阵缓存指令，以得到封装在所述第一矩阵缓存指令中的指令操作码、工作组尺寸、地址偏移量、地址步长、矩阵信息和运算参数，

其中，所述第一排序控制信息包括所述指令操作码、所述工作组尺寸、所述地址偏移量、所述地址步长、所述矩阵信息和所述运算参数，

所述矩阵乘法运算用于实现将第一运算矩阵和第二运算矩阵进行乘法计算，所述第一运算矩阵表示为M*K，所述第二运算矩阵表示为K*N，所述运算参数为K，其中，M、K、N均为正整数，

所述矩阵信息用于指示所述第一矩阵为所述第一运算矩阵或所述第二运算矩阵以及所述第一矩阵是否被转置，

在所述第一矩阵为所述第一运算矩阵的情况下，所述地址步长表示所述第一矩阵的相邻两列数据之间的步长，

在所述第一矩阵为所述第二运算矩阵的情况下，所述地址步长表示所述第一矩阵的相邻两行数据之间的步长。

17.根据权利要求16所述的数据读取电路，其中，所述读取与排序电路包括地址计算子电路、数据排序子电路和读取与缓存子电路，

所述地址计算子电路被配置为：根据所述第一读基地址和所述第一排序控制信息，计算得到与所述第一读基地址对应的多个存储器地址；

所述读取与缓存子电路被配置为：基于所述多个存储器地址，从所述存储器上读取多个待处理数据，其中，所述多个待处理数据表示存储在所述存储器的所述多个存储器地址中的数据；

所述数据排序子电路被配置为：获取所述多个待处理数据；基于所述第一排序控制信息，从所述多个待处理数据中确定所述当前线程束所需的所述P个数据。

18.根据权利要求17所述的数据读取电路，其中，所述第一矩阵包括多个第一数据块，

所述地址计算子电路包括解码模块、循环模块和第一计算模块，

所述解码模块被配置为：根据所述指令操作码、所述工作组尺寸和所述矩阵信息，确定所述多个第一数据块中与所述当前线程束对应的至少一个第一数据块，其中，所述至少一

个第一数据块包括所述多个待处理数据；

所述循环模块被配置为：根据所述至少一个第一数据块的数量和所述运算参数，控制循环状态并输出循环参数；

所述第一计算模块被配置为：基于所述至少一个第一数据块中的每个第一数据块的尺寸、所述循环参数、所述第一读基地址、所述地址步长和所述地址偏移量，计算得到所述多个存储器地址。

19.根据权利要求18所述的数据读取电路，其中，所述第一计算模块包括第一计算子模块和第二计算子模块，

所述第一计算子模块被配置为：基于所述至少一个第一数据块中的每个第一数据块的尺寸、所述循环参数、所述第一读基地址、所述地址步长和所述地址偏移量，计算得到多个首地址，其中，所述至少一个第一数据块中的每个第一数据块的数据排列为多行多列，在所述第一矩阵为所述第一运算矩阵的情况下，所述多个首地址为所述至少一个第一数据块的各列数据中的第一个数据的地址，在所述第一矩阵为所述第二运算矩阵的情况下，所述多个首地址为所述至少一个第一数据块的各行数据的第一个数据的地址；

所述第二计算子模块被配置为：基于所述多个首地址，计算所述多个存储器地址。

20.根据权利要求17所述的数据读取电路，其中，所述数据排序子电路还被配置为：

根据所述第一写基地址和所述第一排序控制信息，计算得到与所述P个数据对应的多个寄存器地址，其中，所述多个寄存器地址为所述多个矢量通用寄存器片上用于写入所述P个数据的地址，所述多个寄存器地址表示执行所述矩阵乘法运算时所述P个数据对应的存储地址；

基于所述第一排序控制信息，确定所述P个数据与所述多个寄存器地址的对应关系，基于所述对应关系，将所述P个数据写入所述多个寄存器地址中。

21.根据权利要求20所述的数据读取电路，其中，在执行根据所述第一写基地址和所述第一排序控制信息，计算得到与所述P个数据对应的多个寄存器地址的步骤时，所述数据排序子电路被配置为：

根据所述第一写基地址、所述指令操作码、所述工作组尺寸和所述运算参数，计算得到所述多个寄存器地址。

22.根据权利要求20所述的数据读取电路，其中，在执行基于所述对应关系，将所述P个数据写入所述多个寄存器地址中的步骤时，所述数据排序子电路被配置为：

响应于所述第一矩阵为所述第一运算矩阵：

基于所述对应关系，直接将所述P个数据写入所述多个寄存器地址中；

响应于所述第一矩阵为所述第二运算矩阵：

获取所述P个数据中的第a个数据，其中，a为正整数；

将所述第a个数据复制D份，其中，D为大于1的正整数；

基于所述对应关系，确定所述多个寄存器地址中与所述第a个数据对应的寄存器地址；

将D份所述第a个数据写入与所述第a个数据对应的寄存器地址中。

23.根据权利要求22所述的数据读取电路，其中，在执行响应于所述第一矩阵为所述第一运算矩阵，基于所述对应关系，直接将所述P个数据写入所述多个寄存器地址中的步骤时，所述数据排序子电路被配置为：

将所述P个数据被划分为与所述多个矢量通用寄存器片一一对应的多个数据组；

直接将所述多个数据组分别写入所述多个矢量通用寄存器片上。

24.根据权利要求15所述的数据读取电路，其中，所述读取与排序电路包括地址计算子电路和读取与缓存子电路，

所述地址计算子电路被配置为：根据所述第一读基地址和所述第一排序控制信息，计算得到与所述第一读基地址对应的多个存储器地址；

所述读取与缓存子电路被配置为：基于所述多个存储器地址，从所述存储器上读取所述P个数据，其中，所述P个数据表示存储在所述存储器的所述多个存储器地址中的数据。

25.根据权利要求15～24任一项所述的数据读取电路，其中，在执行解析所述第一矩阵缓存指令，以获得所述第一读基地址的步骤时，

所述硬件调度电路被配置为：

解析所述第一矩阵缓存指令，以得到所述当前线程束所在的工作组对应的工作组基地址；

获取所述当前线程束对应的基地址偏移量；

基于所述工作组基地址和所述基地址偏移量，得到所述第一读基地址。

26.根据权利要求15～24任一项所述的数据读取电路，其中，在执行解析所述第一矩阵缓存指令，以获得所述第一读基地址的步骤时，

所述硬件调度电路被配置为：

解析所述第一矩阵缓存指令，以得到所述当前线程束所在的工作组对应的工作组基地址；

将所述工作组基地址作为所述第一读基地址。

27.根据权利要求15～24任一项所述的数据读取电路，其中，

所述硬件调度电路还被配置为：

获取与所述当前线程束对应的第二矩阵缓存指令，其中，所述第二矩阵缓存指令用于实现对第二矩阵的数据进行读取；

解析所述第二矩阵缓存指令，以获得与所述当前线程束所需的所述第二矩阵中的Q个数据对应的第二读基地址、第二排序控制信息和第二写基地址，其中，Q为正整数；

所述读取与排序电路还被配置为：

基于所述第二读基地址和所述第二排序控制信息，从所述存储器上读取所述Q个数据；

基于所述第二写基地址和所述第二排序控制信息，按照执行所述矩阵乘法运算时所述Q个数据对应的存储地址将所述Q个数据写入所述单指令多数据流单元的多个矢量通用寄存器片上，

其中，所述当前线程束使用写入所述单指令多数据流单元的多个矢量通用寄存器片上的所述P个数据和所述Q个数据进行所述矩阵乘法运算。

数据读取方法和数据读取电路

技术领域

[0001] 本公开的实施例涉及矩阵运算领域，并且更具体地，涉及一种数据读取方法和数据读取电路。

背景技术

[0002] 图形处理器(GPU,Graphics-Processor-Unit)包括多个计算单元(CU,Compute-Unit)，每个计算单元包括多个单指令多数据流结构和片上存储器。每个单指令多数据流包括一组矢量通用寄存器和算术逻辑单元(ALU,arithmetic and logic unit)。单指令多数据流是GPU中执行并行计算的最小单元，可以通过执行一条指令同时控制多个线程执行相同的操作。因为单指令多数据流具有高度的并行性，所以单指令多数据流被广泛应用于矩阵运算。

发明内容

[0003] 本公开至少一实施例提供一种数据读取方法，应用于矩阵乘法运算，其中，所述数据读取方法包括：获取与当前线程束对应的第一矩阵缓存指令，其中，所述第一矩阵缓存指令用于实现对第一矩阵的数据进行读取；解析所述第一矩阵缓存指令，以获得与所述当前线程束所需的所述第一矩阵中的P个数据对应的第一读基地址、第一排序控制信息和第一写基地址，其中，P为正整数；基于所述第一读基地址和所述第一排序控制信息，从存储器上读取所述P个数据；基于所述第一写基地址和所述第一排序控制信息，按照执行所述矩阵乘法运算时所述P个数据对应的存储地址将P个数据写入与所述当前线程束对应的单指令多数据流单元的多个矢量通用寄存器片上。

[0004] 例如，在本公开一实施例提供的数据读取方法中，解析第一矩阵缓存指令，以获得所述第一排序控制信息，包括：解析所述第一矩阵缓存指令，以得到封装在所述第一矩阵缓存指令中的指令操作码、工作组尺寸、地址偏移量、地址步长、矩阵信息和运算参数，其中，所述第一排序控制信息包括所述指令操作码、所述工作组尺寸、所述地址偏移量、所述地址步长、所述矩阵信息和所述运算参数，所述矩阵乘法运算用于实现将第一运算矩阵和第二运算矩阵进行乘法计算，所述第一运算矩阵表示为M*K，所述第二运算矩阵表示为K*N，所述运算参数为K，其中，M、K、N均为正整数，所述矩阵信息用于指示所述第一矩阵为所述第一运算矩阵或所述第二运算矩阵以及所述第一矩阵是否被转置，在所述第一矩阵为所述第一运算矩阵的情况下，所述地址步长表示所述第一矩阵的相邻两列数据之间的步长，在所述第一矩阵为所述第二运算矩阵的情况下，所述地址步长表示所述第一矩阵的相邻两行数据之间的步长。

[0005] 例如，在本公开一实施例提供的数据读取方法中，基于所述第一读基地址和所述第一排序控制信息，从存储器上读取所述P个数据，包括：根据所述第一读基地址和所述第一排序控制信息，计算得到与所述第一读基地址对应的多个存储器地址；基于所述多个存储器地址，从所述存储器上读取多个待处理数据，其中，所述多个待处理数据表示存储在所

述存储器的所述多个存储器地址中的数据;基于所述第一排序控制信息,从所述多个待处理数据中获取所述当前线程束所需的所述P个数据。

[0006] 例如,在本公开一实施例提供的数据读取方法中,所述第一矩阵包括多个第一数据块,根据所述第一读基地址和所述第一排序控制信息,计算得到与所述第一读基地址对应的多个存储器地址,包括:根据所述指令操作码、所述工作组尺寸和所述矩阵信息,确定所述多个第一数据块中与所述当前线程束对应的至少一个第一数据块,其中,所述至少一个第一数据块包括所述多个待处理数据;根据所述至少一个第一数据块的数量和所述运算参数,控制循环状态并输出循环参数;基于所述至少一个第一数据块中的每个第一数据块的尺寸、所述循环参数、所述第一读基地址、所述地址步长和所述地址偏移量,计算得到所述多个存储器地址。

[0007] 例如,在本公开一实施例提供的数据读取方法中,基于所述至少一个第一数据块中的每个第一数据块的尺寸、所述循环参数、所述第一读基地址、所述地址步长和所述地址偏移量,计算得到所述多个存储器地址,包括:基于所述至少一个第一数据块中的每个第一数据块的尺寸、所述循环参数、所述第一读基地址、所述地址步长和所述地址偏移量,计算得到多个首地址,其中,所述至少一个第一数据块中的每个第一数据块的数据排列为多行多列,在所述第一矩阵为所述第一运算矩阵的情况下,所述多个首地址为所述至少一个第一数据块的各列数据中的第一个数据的地址,在所述第一矩阵为所述第二运算矩阵的情况下,所述多个首地址为所述至少一个第一数据块的各行数据的第一个数据的地址;基于所述多个首地址,计算所述多个存储器地址。

[0008] 例如,在本公开一实施例提供的数据读取方法中,基于所述第一写基地址和所述第一排序控制信息,按照执行所述矩阵乘法运算时所述P个数据对应的存储地址将所述P个数据写入与所述当前线程束对应的单指令多数据流单元的多个矢量通用寄存器片上,包括:根据所述第一写基地址和所述第一排序控制信息,计算得到与所述P个数据对应的多个寄存器地址,其中,所述多个寄存器地址为所述多个矢量通用寄存器片上用于写入所述P个数据的地址,所述多个寄存器地址表示执行所述矩阵乘法运算时所述P个数据对应的存储地址;基于所述第一排序控制信息,确定所述P个数据与所述多个寄存器地址的对应关系,基于所述对应关系,将所述P个数据写入所述多个寄存器地址中。

[0009] 例如,在本公开一实施例提供的数据读取方法中,根据所述第一写基地址和所述第一排序控制信息,计算得到与所述P个数据对应的多个寄存器地址,包括:根据所述第一写基地址、所述指令操作码、所述工作组尺寸和所述运算参数,计算得到所述多个寄存器地址。

[0010] 例如,在本公开一实施例提供的数据读取方法中,基于所述对应关系,将所述P个数据写入所述多个寄存器地址中,包括:响应于所述第一矩阵为所述第一运算矩阵:基于所述对应关系,直接将所述P个数据写入所述多个寄存器地址中;响应于所述第一矩阵为所述第二运算矩阵:获取所述P个数据中的第a个数据,其中,a为正整数;将所述第a个数据复制D份,其中,D为大于1的正整数;基于所述对应关系,确定所述多个寄存器地址中与所述第a个数据对应的寄存器地址;将D份所述第a个数据写入与所述第a个数据对应的寄存器地址中。

[0011] 例如,在本公开一实施例提供的数据读取方法中,响应于所述第一矩阵为所述第一运算矩阵,基于所述对应关系,直接将所述P个数据写入所述多个寄存器地址中,包括:将

所述P个数据被划分为与所述多个矢量通用寄存器片一一对应的多个数据组;直接将所述多个数据组分别写入所述多个矢量通用寄存器片上。

[0012] 例如,在本公开一实施例提供的数据读取方法中,基于所述第一读基地址和所述第一排序控制信息,从存储器上读取所述P个数据,包括:根据所述第一读基地址和所述第一排序控制信息,计算得到与所述第一读基地址对应的多个存储器地址;基于所述多个存储器地址,从所述存储器上读取所述P个数据,其中,所述P个数据表示存储在所述存储器的所述多个存储器地址中的数据。

[0013] 例如,在本公开一实施例提供的数据读取方法中,获取与当前线程束对应的第一矩阵缓存指令,包括:接收与所述当前线程束对应的命令;基于所述命令,读取所述第一矩阵缓存指令。

[0014] 例如,在本公开一实施例提供的数据读取方法中,解析所述第一矩阵缓存指令,以获得所述第一读基地址,包括:解析所述第一矩阵缓存指令,以得到所述当前线程束所在的工作组对应的工作组基地址;获取所述当前线程束对应的基地址偏移量;基于所述工作组基地址和所述基地址偏移量,得到所述第一读基地址。

[0015] 例如,在本公开一实施例提供的数据读取方法中,解析所述第一矩阵缓存指令,以获得所述第一读基地址,包括:解析所述第一矩阵缓存指令,以得到所述当前线程束所在的工作组对应的工作组基地址;将所述工作组基地址作为所述第一读基地址。

[0016] 例如,本公开一实施例提供的数据读取方法还包括:获取与所述当前线程束对应的第二矩阵缓存指令,其中,所述第二矩阵缓存指令用于实现对第二矩阵的数据进行读取;解析所述第二矩阵缓存指令,以获得与所述当前线程束所需的所述第二矩阵中的Q个数据对应的第二读基地址、第二排序控制信息和第二写基地址,其中,Q为正整数;基于所述第二读基地址和所述第二排序控制信息,从所述存储器上读取所述Q个数据;基于所述第二写基地址和所述第二排序控制信息,按照执行所述矩阵乘法运算时所述Q个数据对应的存储地址将所述Q个数据写入所述单指令多数据流单元的多个矢量通用寄存器片上,其中,所述当前线程束使用写入所述单指令多数据流单元的多个矢量通用寄存器片上的所述P个数据和所述Q个数据进行所述矩阵乘法运算。

[0017] 本公开一实施例提供一种数据读取电路,应用于矩阵乘法运算,包括:硬件调度电路和读取与排序电路,其中,所述硬件调度电路和所述读取与排序电路耦接,所述硬件调度电路被配置为:获取与当前线程束对应的第一矩阵缓存指令,其中,所述第一矩阵缓存指令用于实现对第一矩阵的数据进行读取;解析所述第一矩阵缓存指令,以获得与所述当前线程束所需的所述第一矩阵中的P个数据对应的第一读基地址、第一排序控制信息和第一写基地址,其中,P为正整数;所述读取与排序电路被配置为:基于所述第一读基地址和所述第一排序控制信息,从存储器上读取所述P个数据;基于所述第一写基地址和所述第一排序控制信息,按照执行所述矩阵乘法运算时所述P个数据对应的存储地址将所述P个数据写入与所述当前线程束对应的单指令多数据流单元的多个矢量通用寄存器片上。

[0018] 例如,在本公开一实施例提供的数据读取电路中,在执行解析所述第一矩阵缓存指令,以获得所述第一排序控制信息的步骤时,所述硬件调度电路被配置为:解析所述第一矩阵缓存指令,以得到封装在所述第一矩阵缓存指令中的指令操作码、工作组尺寸、地址偏移量、地址步长、矩阵信息和运算参数,其中,所述第一排序控制信息包括所述指令操作码、

所述工作组尺寸、所述地址偏移量、所述地址步长、所述矩阵信息和所述运算参数，所述矩阵乘法运算用于实现将第一运算矩阵和第二运算矩阵进行乘法计算，所述第一运算矩阵表示为M*K，所述第二运算矩阵表示为K*N，所述运算参数为K，其中，M、K、N均为正整数，所述矩阵信息用于指示所述第一矩阵为所述第一运算矩阵或所述第二运算矩阵以及所述第一矩阵是否被转置，在所述第一矩阵为所述第一运算矩阵的情况下，所述地址步长表示所述第一矩阵的相邻两列数据之间的步长，在所述第一矩阵为所述第二运算矩阵的情况下，所述地址步长表示所述第一矩阵的相邻两行数据之间的步长。

[0019] 例如，在本公开一实施例提供的数据读取电路中，所述读取与排序电路包括地址计算子电路、数据排序子电路和读取与缓存子电路，所述地址计算子电路被配置为：根据所述第一读基地址和所述第一排序控制信息，计算得到与所述第一读基地址对应的多个存储器地址；所述读取与缓存子电路被配置为：基于所述多个存储器地址，从所述存储器上读取多个待处理数据，其中，所述多个待处理数据表示存储在所述存储器的所述多个存储器地址中的数据；所述数据排序子电路被配置为：获取所述多个待处理数据；基于所述第一排序控制信息，从所述多个待处理数据中确定所述当前线程束所需的所述P个数据。

[0020] 例如，在本公开一实施例提供的数据读取电路中，所述第一矩阵包括多个第一数据块，所述地址计算子电路包括解码模块、循环模块和第一计算模块，所述解码模块被配置为：根据所述指令操作码、所述工作组尺寸和所述矩阵信息，确定所述多个第一数据块中与所述当前线程束对应的至少一个第一数据块，其中，所述至少一个第一数据块包括所述多个待处理数据；所述循环模块被配置为：根据所述至少一个第一数据块的数量和所述运算参数，控制循环状态并输出循环参数；所述第一计算模块被配置为：基于所述至少一个第一数据块中的每个第一数据块的尺寸、所述循环参数、所述第一读基地址、所述地址步长和所述地址偏移量，计算得到所述多个存储器地址。

[0021] 例如，在本公开一实施例提供的数据读取电路中，所述第一计算模块包括第一计算子模块和第二计算子模块，所述第一计算子模块被配置为：基于所述至少一个第一数据块中的每个第一数据块的尺寸、所述循环参数、所述第一读基地址、所述地址步长和所述地址偏移量，计算得到多个首地址，其中，所述至少一个第一数据块中的每个第一数据块的数据排列为多行多列，在所述第一矩阵为所述第一运算矩阵的情况下，所述多个首地址为所述至少一个第一数据块的各列数据中的第一个数据的地址，在所述第一矩阵为所述第二运算矩阵的情况下，所述多个首地址为所述至少一个第一数据块的各行数据的第一个数据的地址；所述第二计算子模块被配置为：基于所述多个首地址，计算所述多个存储器地址。

[0022] 例如，在本公开一实施例提供的数据读取电路中，所述数据排序子电路还被配置为：根据所述第一写基地址和所述第一排序控制信息，计算得到与所述P个数据对应的多个寄存器地址，其中，所述多个寄存器地址为所述多个矢量通用寄存器片上用于写入所述P个数据的地址，所述多个寄存器地址表示执行所述矩阵乘法运算时所述P个数据对应的存储地址；基于所述第一排序控制信息，确定所述P个数据与所述多个寄存器地址的对应关系，基于所述对应关系，将所述P个数据写入所述多个寄存器地址中。

[0023] 例如，在本公开一实施例提供的数据读取电路中，在执行根据所述第一写基地址和所述第一排序控制信息，计算得到与所述P个数据对应的多个寄存器地址的步骤时，所述数据排序子电路被配置为：根据所述第一写基地址、所述指令操作码、所述工作组尺寸和所

述运算参数，计算得到所述多个寄存器地址。

[0024] 例如，在本公开一实施例提供的数据读取电路中，在执行基于所述对应关系，将所述P个数据写入所述多个寄存器地址中的步骤时，所述数据排序子电路被配置为：响应于所述第一矩阵为所述第一运算矩阵：基于所述对应关系，直接将所述P个数据写入所述多个寄存器地址中；响应于所述第一矩阵为所述第二运算矩阵：获取所述P个数据中的第a个数据，其中，a为正整数；将所述第a个数据复制D份，其中，D为大于1的正整数；基于所述对应关系，确定所述多个寄存器地址中与所述第a个数据对应的寄存器地址；将D份所述第a个数据写入与所述第a个数据对应的寄存器地址中。

[0025] 例如，在本公开一实施例提供的数据读取电路中，在执行响应于所述第一矩阵为所述第一运算矩阵，基于所述对应关系，直接将所述P个数据写入所述多个寄存器地址中的步骤时，所述数据排序子电路被配置为：将所述P个数据被划分为与所述多个矢量通用寄存器片一一对应的多个数据组；直接将所述多个数据组分别写入所述多个矢量通用寄存器片上。

[0026] 例如，在本公开一实施例提供的数据读取电路中，所述读取与排序电路包括地址计算子电路和读取与缓存子电路，所述地址计算子电路被配置为：根据所述第一读基地址和所述第一排序控制信息，计算得到与所述第一读基地址对应的多个存储器地址；所述读取与缓存子电路被配置为：基于所述多个存储器地址，从所述存储器上读取所述P个数据，其中，所述P个数据表示存储在所述存储器的所述多个存储器地址中的数据。

[0027] 例如，在本公开一实施例提供的数据读取电路中，在执行解析所述第一矩阵缓存指令，以获得所述第一读基地址的步骤时，所述硬件调度电路被配置为：解析所述第一矩阵缓存指令，以得到所述当前线程束所在的工作组对应的工作组基地址；获取所述当前线程束对应的基地址偏移量；基于所述工作组基地址和所述基地址偏移量，得到所述第一读基地址。

[0028] 例如，在本公开一实施例提供的数据读取电路中，在执行解析所述第一矩阵缓存指令，以获得所述第一读基地址的步骤时，所述硬件调度电路被配置为：解析所述第一矩阵缓存指令，以得到所述当前线程束所在的工作组对应的工作组基地址；将所述工作组基地址作为所述第一读基地址。

[0029] 例如，在本公开一实施例提供的数据读取电路中，所述硬件调度电路还被配置为：获取与所述当前线程束对应的第二矩阵缓存指令，其中，所述第二矩阵缓存指令用于实现对第二矩阵的数据进行读取；解析所述第二矩阵缓存指令，以获得与所述当前线程束所需的所述第二矩阵中的Q个数据对应的第二读基地址、第二排序控制信息和第二写基地址，其中，Q为正整数；所述读取与排序电路还被配置为：基于所述第二读基地址和所述第二排序控制信息，从所述存储器上读取所述Q个数据；基于所述第二写基地址和所述第二排序控制信息，按照执行所述矩阵乘法运算时所述Q个数据对应的存储地址将所述Q个数据写入所述单指令多数据流单元的多个矢量通用寄存器片上，其中，所述当前线程束使用写入所述单指令多数据流单元的多个矢量通用寄存器片上的所述P个数据和所述Q个数据进行所述矩阵乘法运算。

附图说明

[0030] 为了更清楚地说明本公开实施例的技术方案，下面将对实施例的附图作简单地介绍，显而易见地，下面描述中的附图仅仅涉及本公开的一些实施例，而非对本公开的限制。

[0031] 图1为一种线程网格的示意图；

[0032] 图2A～2D为本公开一实施例提供的一种矩阵乘法运算的计算过程的示意图；

[0033] 图3A为本公开一实施例提供的在矩阵乘法运算中矩阵分块的示意图；

[0034] 图3B为本公开一实施例提供的一种子矩阵块的数据和四个线程束之间的映射方式的示意图；

[0035] 图3C为本公开一实施例提供的另一种子矩阵块的数据和四个线程束之间的映射方式的示意图；

[0036] 图3D为一种矩阵的数据在矢量通用寄存器片中的排列方式的示意图；

[0037] 图3E～3H为进行矩阵乘法运算时矢量通用寄存器片中的数据排列方式的示意图；

[0038] 图3I为单精度矩阵乘法运算的一种计算流程图；

[0039] 图4为本公开一些实施例提供的一种数据读取方法的示意性流程图；

[0040] 图5为本公开一些实施例提供的一种矩阵缓存指令的格式示意图；

[0041] 图6A～6C为本公开一些实施例提供的矩阵乘法运算的不同映射方式的示意图；

[0042] 图7A为本公开一些实施例提供的单个线程束对应的数据块和寄存器地址之间的关系示意图；

[0043] 图7B为本公开一些实施例提供的单个线程束对应的数据块的各个数据的示意图；

[0044] 图8A～8D示出了图7B所示的数据块A0、数据块A1、数据块B0和数据块B1在矢量通用寄存器片中的数据排列方式的示意图；

[0045] 图9为本公开一些实施例提供的一种数据读取电路的示意图；

[0046] 图10为本公开一些实施例提供的一种地址计算子电路的结构示意图。

具体实施方式

[0047] 为了使得本公开实施例的目的、技术方案和优点更加清楚，下面将结合本公开实施例的附图，对本公开实施例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的一部分实施例，而不是全部的实施例。基于所描述的本公开的实施例，本领域普通技术人员在无需创造性劳动的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。

[0048] 除非另外定义，本公开使用的技术术语或者科学术语应当为本公开所属领域内具有一般技能的人士所理解的通常意义。本公开中使用的“第一”、“第二”以及类似的词语并不表示任何顺序、数量或者重要性，而只是用来区分不同的组成部分。“包括”或者“包含”等类似的词语意指出现该词前面的元件或者物件涵盖出现在该词后面列举的元件或者物件及其等同，而不排除其他元件或者物件。“连接”或者“相连”等类似的词语并非限定于物理的或者机械的连接，而是可以包括电性的连接，不管是直接的还是间接的。“上”、“下”、“左”、“右”等仅用于表示相对位置关系，当被描述对象的绝对位置改变后，则该相对位置关系也可能相应地改变。

[0049] 下面简单介绍GPU中关于矩阵乘法运算的相关内容。

[0050] 软硬件交互的交互接口模块(SPI:Software-Program-Interface),用于软件向硬件发送命令,作为硬件和软件之间的接口模块,SPI可以将一些初始化的信息传给计算单元(CU),还可以将准备好的命令以工作组(Workgroup)的方式发送给调度模块(SQ,Sequencer:用于计算任务调度的硬件模块,可以实现取指、译码、内存读写、流控等功能)。SQ接收到新的工作组的命令之后就可以进行指令的读取、译码和执行等操作。当执行的指令表示需要进行算术逻辑计算(通过单指令多数据流中的ALU实现,ALU能实现多组算术运算和逻辑运算)时,SQ就将相应的操作就发送给CU;当执行的指令表示需要从存储器(片外存储器,例如,DDR SDRAM(Double-Data-Rate Double Data Rate))读取数据时,SQ就将相应的命令就发送给地址计算硬件模块。ALU可以执行多条指令计算出需要被读取的数据在存储器上的地址,CU会将该地址发送给地址计算硬件模块,地址计算硬件模块会对该地址进行再处理,然后发送到存储系统(memory system,包括缓存器cache)。存储系统基于该地址控制线程从存储器上读取数据,并将读取的数据返回给数据重组硬件模块,数据重组硬件模块对该读取的数据进行格式转换,然后将格式转换后的数据返回给CU并存储到单指令多数据流的多个矢量通用寄存器片中。

[0051] 例如,每个矢量通用寄存器片包括多个矢量通用寄存器,例如,256个矢量通用寄存器,例如,每个矢量通用寄存器片包括的256个矢量通用寄存器的地址分别为v0～v255。

[0052] 当前的GPU架构中,一个计算单元(CU)包括四个单指令多数据流,每个单指令多数据流按照线程束(wave,单指令多数据流上执行的一个单位的并行计算的线程数量,一个线程束可以包括64个线程(Thread,Thread是GPU中执行的最小粒度))的粒度进行并行的计算。通常,一个CU中的四个单指令多数据流依次序轮流占用地址计算硬件模块和数据重组硬件模块,以进行的数据的读写操作。每个单指令多数据流的ALU只能访问该单指令多数据流中的矢量通用寄存器,不同的单指令多数据流之间共享数据是通过CU内的一个片上存储器来实现。

[0053] 图1为一种线程网格的示意图,如图1所示,GPU在分配计算任务时,一个任务可以用一个线程网格(grid,一次计算任务的大小)来代表,grid又可以被划分成多个工作组(workgroup,OpenCL(Open-Computing-Language)里定义的最小工作组),一个工作组只能分到一个硬件的CU上去计算。一个工作组又可以被划分为多个wave(一个wave包括64个线程),一个wave在一个单指令多数据流上执行。

[0054] 在当前Rocm(一个开源的具有高性能和超大规模集群GPU通用计算的平台)中,一般的矩阵乘法运算(x-general-matrix-multiplication,xgemm,x可以表示s:single precision,单精度,此时,表示为sgemm(single-precision-general-matrix-multiplication,即单精度的数据的矩阵乘法运算),x也可以表示d:double precision,双精度,此时,表示为dgemm(double-precision-general-matrix-multiplication,即双精度的数据的矩阵乘法运算))可以用下面的示例来进行说明。

[0055] 例如,对于矩阵乘法运算:C(M,N)＝A(M,K)*B(K,N)。矩阵C(M行N列)由矩阵A(M行K列)与矩阵B(K行N列)相乘得到。假如,M＝8,N＝4,K＝4,那么,在硬件电路中,该矩阵乘法运算的计算过程可以如图2A～2D所示。

[0056] 如图2A所示,第一次计算用矩阵A的第一列A[i:1]的数据(包括8个数据)乘以矩阵B的第一行B[1:j]的数据(包括4个数据),从而得到矩阵C中的每个元素(数据)的第一次部

分值C1,该第一次部分值C1作为矩阵C中的各个元素的第一次累加结果。

[0057] 如图2B所示,第二次计算用矩阵A的第二列A[i:2]的数据乘以矩阵B的第二行B[2:j]的数据,从而得到矩阵C中的每个元素的第二次部分值C2,第二次部分值C2与第一次累加结果,即第一次部分值C1,进行累加,以得到矩阵C中的各个元素的第二次累加结果。

[0058] 如图2C所示,第三次计算用矩阵A的第三列A[i:3]的数据乘以矩阵B的第三行B[3:j]的数据,从而得到矩阵C中的每个元素的第三次部分值C3,第三次部分值C3与第二次累加结果进行累加,以得到矩阵C中的各个元素的第三次累加结果。

[0059] 如图2D所示,第四次计算用矩阵A的第四列A[i:4]的数据乘以矩阵B的第四行B[4:j]的数据,从而得到矩阵C中的每个元素的第四次部分值C4,第四次部分值C4与第三累加结果进行累加,以得到矩阵C中的各个元素的第四次累加结果。该第四次累加结果即为该矩阵乘法运算的最终结果,即最终矩阵C中的各个元素的值。

[0060] 例如,在图2A～2D中,i和j均为正整数,且i的范围为1～8,j的范围为1～4。

[0061] 在利用硬件电路进行矩阵乘法运算之前,各个矩阵(例如,上述矩阵A和矩阵B)的数据需要从存储器读到硬件电路的矢量通用寄存器中。通常,可以利用数据读取指令buffer_load_dowrd/dword*2/dowrd*4(dword这里表示32比特),根据已经算出来的矩阵中的各个数据在存储器上的地址,将矩阵的数据读取到矢量通用寄存器中。例如,当GPU需要执行的矩阵乘法运算为:

[0062] C(128*128)＝A(128*4)*B(4*128)。

[0063] 假设对矩阵C进行分块的参数为:MacroTile:32*32;WorkGroup Shape:16*16;ThreadTile:2*2。MacroTile表示矩阵C经过分块以后每个工作组(workgroup)需要承担计算的子矩阵块的尺寸。WorkGroup Shape表示工作组(workgroup)根据C矩阵分块后的形状所对应的工作组的形状。ThreadTile表示工作组(workgroup)内每个线程(thread)对应的矩阵C的元素个数。一般而言,MacroTile＝WorkGroup Shape*ThreadTile。

[0064] 由此,如图3A所示,矩阵C可以被划分为16个子矩阵块,每个子矩阵块的尺寸为32*32。

[0065] 对于矩阵C的每个子矩阵块(其尺寸为32*32),每个子矩阵块所需的矩阵A的数据为对应列方向上的矩阵A的数据块,每个子矩阵块所需的矩阵B的数据为对应行方向上的矩阵B的数据块。例如,如图3A所示,对于矩阵C中的最左上角的子矩阵块C00,子矩阵块C00所需的矩阵A的数据块为矩阵A的第一个数据块(该数据块的尺寸为32*4),所需的矩阵B的数据块为矩阵B的第一个数据块(该数据块的尺寸为4*32)。下面对矩阵C的子矩阵块C00以及对应矩阵A和B的相关数据块单独进行说明。

[0066] 例如,假设矩阵A是列主序类型的矩阵,矩阵B是行主序类型的矩阵,即矩阵A的数据在存储器中是按照列方向进行存储的,矩阵A中同一列的数据在存储器中的地址是连续的,矩阵B的数据在存储器中是按照行方向进行存储的,矩阵B中同一行的数据在存储器中的地址是连续的。矩阵C的每个子矩阵块在程序上被映射到了一个工作组上(workgroup),该工作组的尺寸WorkGroup Shape为16*16,即工作组包括16*16＝256个线程(thread),即4个wave(一个wave有64个线程)。

[0067] 对于单精度的数据而言,采用buffer_load_dword指令,一个wave可以读取矩阵A的尺寸为32*4的数据块的连续两列数据或者矩阵B的尺寸为4*32的数据块的连续两行数

据,即在同一个时钟周期内,采用两个wave就可以读回所需的矩阵A的尺寸为32*4的数据块或者矩阵B的尺寸为4*32的数据块;而采用buffer_load_dword*2指令,一个wave可以读取矩阵A的尺寸为32*4的数据块的连续4列数据或者矩阵B的尺寸为4*32的数据块的连续4行数据,即在同一个时钟周期内,用一个wave就可以读回所需的矩阵A的尺寸为32*4的数据块或者矩阵B的尺寸为4*32的数据块。对于双精度的数据而言,采用buffer_load_dword指令,一个wave可以读取矩阵A的尺寸为32*4的数据块的一列数据或者矩阵B的尺寸为4*32的数据块的一行数据,即在同一个时钟周期内,同一个工作组内的4个wave可以将所需的矩阵A的尺寸为32*4的数据块或者矩阵B的尺寸为4*32的数据块读取回来。

[0068] 在不同尺寸的矩阵乘法运算中,映射到工作组上的子矩阵块的尺寸是程序可任意划分的,为了最大化利用一个wave读取数据的最大带宽,通常会根据不同的情况用到buffer_load_dword、buffer_load_dword*2或者buffer_load_dword*4指令,需要说明的是,每个线程读取回来的数据可能并不是这个线程在做矩阵乘法运算时所用的数据,甚至在某些情况下会有一个wave计算所用的数据是同一个工作组内其他wave读回来的情况。

[0069] 图3B示出了一种子矩阵块的数据和四个线程束之间的映射方式的示意图;图3C示出了另一种子矩阵块的数据和四个线程束之间的映射方式的示意图。

[0070] 以一个子矩阵块C00(尺寸为32*32)被划分到包括四个线程束的工作组为例,在进行矩阵乘法运算时,子矩阵块的数据和四个线程束之间的对应关系可以有且不限于图3B和图3C示出的映射方式。

[0071] 例如,如图3B和图3C所示,四个线程束分别为线程束wave0、线程束wave1、线程束wave2和线程束wave3,子矩阵块C00对应矩阵A的数据块A0和数据块A1(即尺寸为32*4的数据块被划分为两个16*4的数据块)以及矩阵B的数据块B0和数据块B1(即尺寸为4*32的数据块被划分为两个4*16的数据块)。数据块A0的尺寸和数据块A1的尺寸均为16*4;数据块B0的尺寸和数据块B1的尺寸均为4*16。

[0072] 例如,在图3B所示的映射方式中,wave0对应矩阵A的数据块A0和数据块A1的数据以及矩阵B的数据块B0的第1列至第4列的数据和数据块B1的第1列至第4列的数据,wave1对应矩阵A的数据块A0和数据块A1的数据以及矩阵B的数据块B0的第5列至第8列的数据和数据块B1的第5列至第8列的数据,wave2对应矩阵A的数据块A0和数据块A1的数据以及矩阵B的数据块B0的第9列至第12列的数据和数据块B1的第9列至第12列的数据,wave3对应矩阵A的数据块A0和数据块A1的数据以及矩阵B的数据块B0的第13列至第16列的数据和数据块B1的第13列至第16列的数据。

[0073] 例如,在图3C所示的映射方式中,wave0对应矩阵A的数据块A0的数据以及矩阵B的数据块B0的数据,wave1对应矩阵A的数据块A0的数据以及矩阵B的数据块B1的数据,wave2对应矩阵A的数据块A1的数据以及矩阵B的数据块B0的数据,wave3对应矩阵A的数据块A1的数据以及矩阵B的数据块B1的数据。

[0074] 如图3B和图3C所示,在进行矩阵乘法运算时,不同的映射方式中每个线程束对应到子矩阵块C00的内部区域是不同的,进而每个线程束对应矩阵A和矩阵B中的数据也是不同的。在矩阵数据的读取阶段和矩阵运算阶段,线程束内的每个线程所对应的数据不一定是一致的,有可能计算时线程所用的数据是其他线程束或者其他线程读取回来的。

[0075] 根据上面所述,在进行矩阵乘法运算时,每个线程对应的数据可能不是这个线程

自己读取回来的,反映到硬件中则为矩阵的数据读取到计算之间,数据在矢量通用寄存器中的排列方式是不同的,所以在进行矩阵运算时,需要按照矩阵运算时数据的排列方式对数据进行重新排列,然后将重排之后的数据写入矢量通用寄存器中。以上述数据块A0和数据块B0为例,利用buffer_load_dword*指令读回的数据在矢量通用寄存器中的排列方式如图3D所示。

[0076] 例如,每个单指令多数据流包括四个矢量通用寄存器片ROW0～ROW3(图中示出为row0、row1、row2、row3),每个矢量通用寄存器片例如可以包括256个矢量通用寄存器,在图3D中,在数据读取阶段,ROW0存储数据块A0的第一列数据A00～Af0和数据块B0的第一行的第1个至第4个数据B00～B03;ROW1存储数据块A0的第二列数据A01～Af1和数据块B0的第二行的第1个至第4个数据B10～B13;ROW2存储数据块A0的第三列数据A02～Af2和数据块B0的第三行的第1个至第4个数据B20～B23;ROW3存储数据块A0的第四列数据A03～Af3和数据块B0的第四行的第1个至第4个数据B30～B33。

[0077] 在现有技术中,矩阵的数据通过工作组中的若干个wave从存储器读取并写入矢量通用寄存器中,然后通过片上存储器指令将读回来的数据从矢量通用寄存器读出再写到片上存储器中,然后再次通过片上存储器指令操作的方式将数据从片上存储器读出并按照矩阵乘法运算时所需要的格式写回到矢量通用寄存器。最终进行矩阵乘法运算时的矢量通用寄存器中的数据排列方式如图3E～3H所示。

[0078] 如图3E所示,在矩阵运算阶段,ROW0存储数据块A0的四列数据和数据块B0的第一列的4个数据B00～B30,例如,ROW0的v[0:1]存储数据块A0的第一列数据A00～Af0,ROW0的v[2:3]存储数据块A0的第二列数据A01～Af1,ROW0的v[4:5]存储数据块A0的第三列数据A02～Af2,ROW0的v[6:7]存储数据块A0的第四列数据A03～Af3,ROW0的v[80:81]存储数据块B0的第一列的第1个数据B00,ROW0的v[82:83]存储数据块B0的第一列的第2个数据B10,ROW0的v[84:85]存储数据块B0的第一列的第3个数据B20,ROW0的v[86:87]存储数据块B0的第一列的第4个数据B30。

[0079] 在矩阵运算阶段,如图3F所示,ROW1存储数据块A0的四列数据和数据块B0的第二列的4个数据B01～B31,如图3G所示,ROW2存储数据块A0的四列数据和数据块B0的第三列的4个数据B02～B32,如图3H所示,ROW3存储数据块A0的四列数据和数据块B0的第四列的4个数据B03～B33。

[0080] 下面简单说明现有技术中,计算数据在存储器上的地址的过程。

[0081] 在现有的技术方案中,在进行矩阵数据的读取时,线程对应的数据的地址的计算是根据线程所在的工作组在整个线程网格中的位置,再结合线程在其所在的工作组中的位置计算出来的,该地址的计算通常需要若干条整数指令在计算单元里计算得出。例如,线程网格的尺寸为:宽＝X,高＝Y,单位是工作组;工作组的尺寸为:宽＝x,高＝y,单位是线程;工作组在线程网格中的坐标为(tgid_x,tgid_y);工作组中的某个线程在工作组中的坐标为(tid_x,tid_y),那么,该线程在工作组中的地址表示为:

[0082] thread_id_in_group＝tid_y*x+tid_x。

[0083] 该线程所在的工作组在线程网格中的地址表示为:

[0084] thread_group_id＝tgid_y*X+tgid_x。

[0085] 从而,该线程对应的数据的地址表示为:thread_id＝thread_group_id*x*y+

thread_id_in_group。

[0086] 上述地址计算对应的指令序列为：

[0087] //初始化工作组的基本信息，并将初始化的值放入单指令多数据流的标量通用寄存器片(SGPR)上

[0088] //s[0:1]:buffer的描述符信息在存储器中的地址

[0089] //s2:X

[0090] //s3:Y

[0091] //s5:x

[0092] //s6:y

[0093] //s12:tgid_x

[0094] //s13:tgid_y

[0095] //v0:tid_x

[0096] //v1:tid_y,v0和v1表示在单指令多数据流的矢量通用寄存器的地址

[0097] //将buffer的描述符信息读取到标量通用寄存器的s[20:23]

[0098] s_load_dwordx4 s[20:23],s[0:1],0x0(0x表示十六进制数据)

[0099] s_waitcnt 0

[0100] //计算thread_id_in_group:v8＝tid_y*x+tid_x

[0101] v_mad_u32_u24 v8,v1,s5,v0

[0102] //计算thread_group_id:s26＝tgid_y*X+tgid_x

[0103] s_mul_i32 s26,s13,s2

[0104] s_add_i32 s26,s26,s12

[0105] //计算thread_id:v9＝thread_group_id*x*y+thread_id_in_group

[0106] //s30＝x*y,可以认为是工作组的基地址(base address)

[0107] s_mul_i32 s30,s5,s6

[0108] v_mov_b32 v9,s26

[0109] v_mad_u32_u24 v9,v9,s30,v8

[0110] 例如，s[0:1]、s2、s3、s5、s6、s12、s13表示在单指令多数据流的SGPR的地址。

[0111] 以上汇编代码中使用的thread group就是上述workgroup，用thread group第一个线程的地址代替thread group的首地址是为了简化了地址计算的过程以方便说明。

[0112] 在当前的Rocm中，xgemm的计算流程如图3I所示，在图3I中，第一步：Addr calculation，即计算出对应矩阵A和矩阵B的数据在存储器上的地址。第二步：用指令buffer_load_dword*4将矩阵A和矩阵B的数据从存储器读回到矢量通用寄存器中。vmcnt(0)指令表示等待矩阵A的所有数据和矩阵B的所有数据均读取完毕再进行下一步操作。第三步：采用片上存储器指令，比如ds_write_b128，将矩阵A和矩阵B的数据从矢量通用寄存器读出并写到片上存储器中。因为在一个工作组(workgroup)内，不同wave计算的子矩阵块所需的数据有可能是其他wave读取回来的，所以这里需要等待一个工作组内所有的wave都将数据写入到片上存储器中才能执行下一步，例如，通过s_barrier指令和lgckmcnt(0)指令进行控制。第四步：采用片上存储器指令，比如ds_read_b128，将矩阵A和矩阵B的数据从片上存储器中读出，并按照矩阵乘法运算所需的格式写入到矢量通用寄存器中。lgckmcnt

(0)指令表示等待矩阵A的所有数据和矩阵B的所有数据均从片上存储器中读出再进行下一步操作。第五步,利用FMA指令(Fused-Multiply-Add,即积和熔加运算),例如,v_fma_f64,执行矩阵相乘计算。s_setprio 0/1指令表示设置后续指令的优先级。第六步,如果当前子矩阵块的矩阵乘法运算需要若干步的FMA计算操作,判断当前子矩阵块的运算是否完毕,即“unroll done?”,若没有完毕,则循环第四步至第五步直到当前子矩阵块的运算执行完毕。第七步,当要计算多个子矩阵块时,则判断当前子矩阵块是否是最后一个子矩阵块,若不是,重复上述第一步至第六步的操作,直到所有子矩阵块均运算完毕。第八步,利用Buffer_store_dword*4指令将计算出的矩阵C写回到存储器中。

[0113] 基于上述对现有的矩阵乘法运算的流程的说明可知,矩阵A和矩阵B的数据从存储器读回到矢量通用寄存器后并不能马上开始矩阵乘法运算的过程,需要利用片上存储器对矩阵A和矩阵B的数据进行重排,这个过程导致了数据准备时间总延迟的增加,同时也增加了片上存储器和矢量通用寄存器的读写次数,消耗了更多的功耗,而且还占用了片上存储器的空间。

[0114] 针对上述方案的不足,本公开的实施例的提供了一种用于矩阵乘法运算的数据读取方法和数据读取电路,以达到减小数据读取总延迟,提高矩阵运算效率的目的,解决数据读取延迟过长,片上存储器读写次数过多,且占用片上存储器空间的问题。

[0115] 该数据读取方法包括:获取与当前线程束对应的第一矩阵缓存指令,其中,第一矩阵缓存指令用于实现对第一矩阵的数据进行读取;解析第一矩阵缓存指令,以获得与当前线程束所需的第一矩阵中的P个数据对应的第一读基地址、第一排序控制信息和第一写基地址,其中,P为正整数;基于第一读基地址和第一排序控制信息,从存储器上读取P个数据;基于第一写基地址和第一排序控制信息,按照执行矩阵乘法运算时P个数据对应的存储地址将P个数据写入与当前线程束对应的单指令多数据流单元的多个矢量通用寄存器片上。

[0116] 基于该数据读取方法,可以将数据从存储器读回时直接按照矩阵乘法运算所需要的排序方式写入到矢量通用寄存器中,从而减小数据准备时间和片上存储器读写次数,减少片上存储器的占用空间,提高矩阵运算效率。

[0117] 下面结合附图对本公开的实施例进行详细说明,但是本公开并不限于这些具体的实施例。为了保持本公开实施例的以下说明清楚且简明,本公开省略了部分已知功能和已知部件的详细说明。

[0118] 图4为本公开一些实施例提供的一种数据读取方法的示意性流程图。

[0119] 例如,数据读取方法应用于矩阵运算,例如,矩阵乘法运算,如图4所示,该数据读取方法包括:

[0120] 步骤S10:获取与当前线程束对应的第一矩阵缓存指令;

[0121] 步骤S20:解析第一矩阵缓存指令,以获得与当前线程束所需的第一矩阵中的P个数据对应的第一读基地址、第一排序控制信息和第一写基地址,其中,P为正整数;

[0122] 步骤S30:基于第一读基地址和第一排序控制信息,从存储器上读取P个数据;

[0123] 步骤S40:基于第一写基地址和第一排序控制信息,按照执行矩阵乘法运算时P个数据对应的存储地址将P个数据写入与当前线程束对应的单指令多数据流单元的多个矢量通用寄存器片上。

[0124] 例如,在一些实施例中,数据读取方法还包括:获取与当前线程束对应的第二矩阵

缓存指令，其中，第二矩阵缓存指令用于实现对第二矩阵的数据进行读取；解析第二矩阵缓存指令，以获得与当前线程束所需的第二矩阵中的Q个数据对应的第二读基地址、第二排序控制信息和第二写基地址，其中，Q为正整数；基于第二读基地址和第二排序控制信息，从存储器上读取Q个数据；基于第二写基地址和第二排序控制信息，按照执行矩阵乘法运算时Q个数据对应的存储地址将Q个数据写入与单指令多数据流单元的多个矢量通用寄存器片上。

[0125] 例如，当前线程束使用写入单指令多数据流单元的多个矢量通用寄存器片上的P个数据和Q个数据进行矩阵乘法运算。

[0126] 例如，通过第一矩阵缓存指令实现对第一矩阵的数据进行读取，通过第二矩阵缓存指令实现对第二矩阵的数据进行读取，然后，基于读取的第一矩阵的数据和第二矩阵的数据，该当前线程束即可进行矩阵乘法运算。

[0127] 对第一矩阵的数据进行读取的过程和对第二矩阵的数据进行读取的过程类似，下面以对第一矩阵的数据进行读取的过程为例对本公开的实施例进行说明，需要注意的是，下面对于读取第一矩阵的数据的描述，在不矛盾的情况下同样适用读取第二矩阵的数据的过程。

[0128] 在本公开实施例中，步骤S10和S20均是由硬件调度电路(后续将描述)实现的，步骤S30和S40均是由读取与排序电路(后续将描述，且包括地址计算子电路、数据排序子电路和读取与缓存子电路)实现的。

[0129] 例如，在步骤S10中，第一矩阵缓存指令用于实现对第一矩阵的数据进行读取。

[0130] 例如，在一些实施例中，步骤S10包括：接收与当前线程束对应的命令；基于命令，读取第一矩阵缓存指令。

[0131] 例如，与当前线程束对应的命令可以通过交互接口模块SPI发送给硬件调度电路，然后硬件调度电路根据该命令读取第一矩阵缓存指令。

[0132] 例如，在步骤S20中，当前线程束所需的P个数据由矩阵映射方式决定，可以为第一矩阵的所有数据，也可以为第一矩阵的部分数据。

[0133] 例如，在一些实施例中，在步骤S20中，解析第一矩阵缓存指令，以获得第一排序控制信息，包括：解析第一矩阵缓存指令，以得到封装在第一矩阵缓存指令中的指令操作码、工作组尺寸、地址偏移量、地址步长、矩阵信息和运算参数。

[0134] 例如，第一排序控制信息包括指令操作码、工作组尺寸、地址偏移量、地址步长、矩阵信息和运算参数。

[0135] 例如，矩阵乘法运算用于实现将第一运算矩阵和第二运算矩阵进行乘法计算，例如，第一运算矩阵可以为列主序类型的矩阵，第二运算矩阵可以为行主序类型的矩阵。在下面的描述中，以第一运算矩阵为列主序类型的矩阵A，第二运算矩阵为行主序类型的矩阵B为例进行说明。

[0136] 例如，若第一矩阵为第一运算矩阵A，则第二矩阵为第二运算矩阵B；若第一矩阵为第二运算矩阵B，则第二矩阵为第一运算矩阵A。

[0137] 例如，若第一运算矩阵表示为M*K，第二运算矩阵表示为K*N，则运算参数为K。M、K、N均为正整数。

[0138] 例如，矩阵信息用于指示第一矩阵为第一运算矩阵或第二运算矩阵以及第一矩阵

是否被转置;在第一矩阵为第一运算矩阵的情况下,地址步长表示第一矩阵的相邻两列数据之间的步长,在第一矩阵为第二运算矩阵的情况下,地址步长表示第一矩阵的相邻两行数据之间的步长。

[0139] 例如,对于列主序类型的第一运算矩阵A的数据块,由于不同数据块的同一列的数据在地址上是连续的,地址偏移量表示:在列方向上,当前数据块相对于前一个数据块的同一个位置的数据之间在地址上的偏移量;对于行主序类型的第二运算矩阵B的数据块,由于不同数据块的同一行的数据在地址上是连续的,地址偏移量表示:在行方向上,当前数据块相对于前一个数据块的同一个位置的数据之间在地址上的偏移量。

[0140] 对于第一矩阵中的相邻的两个数据块,例如,如图3C所示的数据块A0和数据块A1,地址偏移量表示数据块A0的第一行第一列的数据a0的地址和数据块A0的第一行第一列的数据a1的地址之间的偏移量。例如,参见附图7B(后续详细描述),对于第一运算矩阵A中的数据块A0和数据块A1,数据块A1相对于数据块A0的地址偏移量为数据A00的地址和数据A160的地址之间的偏移量。

[0141] 本公开的实施例提供一种新的针对矩阵类型数据的读取指令:矩阵缓存指令(Matrix buffer,MMBUF),基于该矩阵缓存指令封装的信息,该数据读取方法可以实现将数据从存储器读回时直接按照矩阵乘法运算所需要的排序方式写入到矢量通用寄存器的功能。例如,MMBUF指令首先可以根据目标矩阵(例如第一矩阵)的尺寸以及硬件资源的情况,通过灵活的参数配置支持各种矩阵乘法运算时的数据读取,同时数据读回时,可以直接按照矩阵乘法运算时的数据排序方式写进矢量通用寄存器,改进了数据准备的延迟,减少了指令的开销,节省了功耗。通过对MMBUF指令的译码,硬件电路可以自动计算第一运算矩阵和第二运算矩阵中的各个数据块的地址(例如,存储器地址和寄存器地址)。将矩阵的数据读回时,硬件电路可以对读取的数据进行重新排序。

[0142] 例如,第一矩阵缓存指令和第二矩阵缓存指令即为此处描述的MMBUF指令。

[0143] 下面介绍MMBUF指令。

[0144] MMBUF指令的格式如图5所示。

[0145] 例如,如图5所示,该MMBUF指令中第一行的第25位至第31位存储的数据,即1110000,用于表示该指令为MMBUF指令。

[0146] 该MMBUF指令中第一行的第18位至第24位存储OP7,即指令操作码(opcode),下面表1示出了一些指令操作码的示例。

[0147] 表1

[0148]

Opcode	类型	Opcode	类型
0	mmbuf_load1*1_dword	32	mmbuf_load1*1_dword*2
1	mmbuf_load2*2_dword	33	mmbuf_load2*2_dword*2
2	mmbuf_load2*4_dword	34	mmbuf_load2*4_dword*2
3	mmbuf_load4*2_dword	35	mmbuf_load4*2_dword*2
4	mmbuf_load4*4_dword	36	mmbuf_load4*4_dword*2
5	mmbuf_load4*6_dword	37	mmbuf_load4*6_dword*2
6	mmbuf_load4*8_dword	38	mmbuf_load4*8_dword*2
7	mmbuf_load6*4_dword	39	mmbuf_load6*4_dword*2
8	mmbuf_load6*6_dword	40	mmbuf_load6*6_dword*2
9	mmbuf_load6*8_dword	41	mmbuf_load6*8_dword*2
10	mmbuf_load8*2_dword	42	mmbuf_load8*2_dword*2
11	mmbuf_load8*4_dword	43	mmbuf_load8*4_dword*2
12	mmbuf_load8*6_dword	44	mmbuf_load8*6_dword*2
13	mmbuf_load8*8_dword	45	mmbuf_load8*8_dword*2

[0149] 指令操作码中的m*n(例如,1*1、2*2等)代表ThreadTile的信息。

[0150] 该MMBUF指令中第一行的第13位存储的信息AB表示该MMBUF指令对应第一运算矩阵A还是第二运算矩阵B,AB为0则表示该MMBUF指令对应的是第一运算矩阵A的数据,AB为1则表示该MMBUF指令对应的是第二运算矩阵B的数据。由此,当第一矩阵为第一运算矩阵A时,第一矩阵缓存指令封装的信息AB为0;当第一矩阵为第二运算矩阵B时,第一矩阵缓存指令封装的信息AB为1。例如,在一些示例中,第一矩阵为第一运算矩阵A,第二矩阵为第二运算矩阵B,此时,第一矩阵缓存指令封装的信息AB为0,第二矩阵缓存指令封装的信息AB为1。

[0151] 该MMBUF指令中第一行的第12位存储的信息T表示该MMBUF所对应的矩阵是否需要转置,T为0则表示矩阵未被转置,T为1则表示矩阵被转置,例如,若第一运算矩阵A是列主序类型的矩阵,那么第一运算矩阵A对应的MMBUF指令中的T为0;若第二运算矩阵B是行主序类型的矩阵,那么第二运算矩阵B对应的MMBUF指令中的T为1。例如,若第一矩阵为列主序类型的矩阵,则第一矩阵缓存指令中的T为0;若第一矩阵为行主序类型的矩阵,则第一矩阵缓存指令中的T为1。

[0152] 例如,第一排序控制信息包括的矩阵信息包括第一矩阵缓存指令中的信息AB和T。

[0153] 该MMBUF指令中第一行的第0位至第12位存储的信息OFFSET12表示基地址偏移量。

[0154] 该MMBUF指令中第二行的第0位至第7位存储的信息STRIDE表示步长,例如,对于第一运算矩阵A是列主序类型的矩阵,第二运算矩阵B是行主序类型的矩阵的情况,那么,第一运算矩阵A对应的MMBUF指令中的STRIDE代表第一运算矩阵A的相邻两列数据之间的步长,第二运算矩阵B对应的MMBUF指令中的STRIDE代表第二运算矩阵B的相邻两行数据之间的步长。

[0155] 例如,第一排序控制信息包括的地址步长即为第一矩阵缓存指令中的信息STRIDE。

[0156] 该MMBUF指令中第二行的第8位至第15位存储的信息VDATA表示读回来的数据写入矢量通用寄存器的第一个索引地址,即第一写基地址。编译器在用MMBUF指令时寄存器地址的分配是连续的,这样除了第一写基地址之外,第一运算矩阵A或者第二运算矩阵B中的所有数据的寄存器地址(即矢量通用寄存器索引地址)均由硬件电路,例如,地址计算子电路或数据排序子电路,计算出来。

[0157] 封装在MMBUF指令描述域(V#)中的信息包括:

[0158] BaseAddr:表示第一运算矩阵或第二运算矩阵在存储器中的基地址;

[0159] K[2:0]:表示矩阵乘法运算的K值,即运算参数,K可以为1、4、8、16、32、64、128,例如,在图2A至图2D所示的矩阵乘法运算中,K为4;

[0160] WGSHAPE:表示工作组尺寸,即WorkGroup Shape,可以为2*8、2*16、2*32、4*4、4*8、4*16、4*64、8*8、8*16、8*32、16*16、16*32等。

[0161] 需要说明的是,地址偏移量可以由地址计算子电路解析WGSHAPE和opcode以后,判断出需要读取的数据块的个数而自动计算出来的,表2和表3(后续描述)就是计算地址偏移量的一个示例。

[0162] 若以图3A中的矩阵分块和图3C中的映射方式作为示例,MMBUF指令在汇编代码中的语义表现形式可如下所示。

[0163] 每个wave读取第一运算矩阵的一个16*4数据块对应的MMBUF指令表示为:

[0164] VDST SRC0 SRC1 SRC2 MODIFIERS mmbuf_load2*2_dword vdata,stride,srsrc,soffset,AB:0,T:0,WGSHAPE:16*16,k:4

[0165] 每个wave读取第二运算矩阵的一个4*16数据块对应的MMBUF指令表示为:

[0166] VDST SRC0 SRC1 SRC2 MODIFIERS mmbuf_load2*2_dword vdata,stride,srsrc,soffset,AB:1,T:1,WGSHAPE:16*16,k:4

[0167] 其中,srsrc表示MMBUF指令的V#在标量通用寄存器片(SGPR)上的地址,stride、srsrc和soffset的值均存储在SGPR上。

[0168] 在本公开中,MMBUF指令的作用在于:在矩阵数据读写的同时也完成矩阵乘法运算时矩阵数据在矢量通用寄存器中的排序。此时,在执行每个wave对应的MMBUF指令时,读取的数据即为该wave进行矩阵乘法运算时所需的数据。例如,在图3B所示的映射方式中,在进行矩阵乘法运算时,wave0需要数据块A0、数据块A1、数据块B0的部分数据(例如,第一列至第四列)和数据块B1的部分数据(例如,第一列至第四列),那么在读取数据时,当执行wave0对应的第一MMBUF指令(该第一MMBUF指令用于实现对第一运算矩阵A的数据进行读取)时,读取的数据为数据块A0和数据块A1,当执行wave0对应的第二MMBUF指令(该第二MMBUF指令用于实现对第二运算矩阵B的数据进行读取)时,读取的数据为数据块B0和数据块B1,然后,在将数据写入矢量通用寄存器时,选择与wave0对应的数据,即数据块A0、数据块A1、数据块B0的部分数据(例如,第一列至第四列)和数据块B1的部分数据(例如,第一列至第四列),写入矢量通用寄存器。同理,wave1、wave2和wave3与wave0的情况类似。

[0169] 而在图3C所示的映射方式中,在进行矩阵乘法运算时,wave0需要数据块A0和数据块B0,wave1需要数据块A0和数据块B1,wave2需要数据块A1和数据块B0,wave3需要数据块A1和数据块B1,那么在读取数据时,当执行wave0对应的第一MMBUF指令时,读取的数据为数据块A0,当执行wave0对应的第二MMBUF指令时,读取的数据为数据块B0的数据;当执行

wave1对应的第一MMBUF指令时，读取的数据为数据块A0的数据，当执行wave1对应的第二MMBUF指令时，读取的数据为数据块B1的数据；当执行wave2对应的第一MMBUF指令时，读取的数据为数据块A1的数据，当执行wave2对应的第二MMBUF指令时，读取的数据为数据块B0的数据；当执行wave3对应的第一MMBUF指令时，读取的数据为数据块A1的数据，当执行wave3对应的第二MMBUF指令时，读取的数据为数据块B1的数据。下面的说明中以图3C所示的映射方式为例来描述本公开的实施例。

[0170] MMBUF指令通过Opcode、WGSHAPE以及K值可以实现多种矩阵乘法运算方式的组合，如下所示。

[0171] 方式一：Opcode＝1*1，WGSHAPE＝8*8，k＝4：此时，工作组只有一个wave，对应矩阵乘法运算表示为$C_{(8*8)}＝A_{(8*4)}$ x $B_{(4*8)}$。

[0172] 方式二：Opcode＝2*2，WGSHAPE＝16*16，k＝4：此时，工作组包括四个wave，对应矩阵乘法运算表示为$C_{(32*32)}＝A_{(32*4)}$ x $B_{(4*32)}$。方式二的情况如图3C所示，在方式二中，矩阵C即作为子矩阵块，子矩阵块由四个wave执行，第一运算矩阵A被划分为两个数据块，第一运算矩阵A的每个数据块的尺寸为16*4，第二运算矩阵B被划分为两个数据块，第二运算矩阵B的每个数据块的尺寸为4*16。

[0173] 方式三：Opcode＝4*4，WGSHAPE＝16*16，k＝4：此时，工作组有四个wave，对应矩阵乘法运算表示为$C_{(64*64)}＝A_{(64*4)}$ x $B_{(4*64)}$。例如，方式三的情况如图6A所示，在方式三中，矩阵C被划分为四个子矩阵块，该四个子矩阵块分别由四个wave执行，每个子矩阵块的尺寸为32*32，第一运算矩阵A被划分为四个数据块A0～A3，第一运算矩阵A的每个数据块的尺寸为16*4，第二运算矩阵B被划分为四个数据块B0～B3，第二运算矩阵B的每个数据块的尺寸为4*16，每个子矩阵块对应第一运算矩阵A的两个数据块和第二运算矩阵B的两个数据块。

[0174] 方式四：Opcode＝6x6，WGSHAPE＝16*16，k＝4：此时，工作组有四个wave，对应矩阵乘法运算表示为$C_{(96*96)}＝A_{(96*4)}$ x $B_{(4*96)}$。例如，方式四的情况如图6B所示，在方式四中，矩阵C被划分为四个子矩阵块，该四个子矩阵块分别由四个wave执行，每个子矩阵块的尺寸为48*48，第一运算矩阵A被划分为六个数据块，第一运算矩阵A的每个数据块的尺寸为16*4，第二运算矩阵B被划分为六个数据块，第二运算矩阵B的每个数据块的尺寸为4*16，每个子矩阵块对应第一运算矩阵A的三个数据块和第二运算矩阵B的三个数据块。

[0175] 方式五：Opcode＝8*8，WGSHAPE＝16*16，k＝4：此时，工作组有四个wave对应矩阵乘法运算表示为$C_{(128*128)}＝A_{(128*4)}$ x $B_{(4*128)}$。例如，方式五的情况如图6C所示，在方式五中，矩阵C被划分为四个子矩阵块，该四个子矩阵块分别由四个wave执行，每个子矩阵块的尺寸为64*64，第一运算矩阵A被划分为八个数据块，第一运算矩阵A的每个数据块的尺寸为16*4，第二运算矩阵B被划分为八个数据块，第二运算矩阵B的每个数据块的尺寸为4*16，每个子矩阵块对应第一运算矩阵A的四个数据块和第二运算矩阵B的四个数据块。

[0176] 需要说明的是，MMBUF不仅限于在上述具体示例中使用。

[0177] 例如，在一些实施例中，在步骤S20中，解析第一矩阵缓存指令，以获得第一读基地址，包括：解析第一矩阵缓存指令，以得到当前线程束所在的工作组对应的工作组基地址；将工作组基地址作为第一读基地址。

[0178] 例如，工作组基地址即为第一矩阵缓存指令中的BaseAddr。

[0179] 例如，对于图3B所示的映射方式，若第一矩阵为第一运算矩阵A，即第一矩阵缓存

指令用于实现对第一运算矩阵A的数据进行读取时，对于wave0～wave3，第一矩阵缓存指令中的BaseAddr均为数据a0的地址，即wave0～wave3对应的工作组基地址(即第一读基地址)均为数据a0的地址；若第一矩阵为第二运算矩阵B，即第一矩阵缓存指令用于实现对第二运算矩阵B的数据进行读取时，对于wave0～wave3，第一矩阵缓存指令中的BaseAddr均为数据b0的地址，即wave0～wave3对应的工作组基地址(即第一读基地址)均为数据b0的地址。

[0180] 例如，在另一些实施例中，在步骤S20中，解析第一矩阵缓存指令，以获得第一读基地址，包括：解析第一矩阵缓存指令，以得到当前线程束所在的工作组对应的工作组基地址；获取当前线程束对应的基地址偏移量；基于工作组基地址和基地址偏移量，得到第一读基地址。

[0181] 例如，基地址偏移量表示不同数据块的第一行第一列的数据的地址之间的偏移量。

[0182] 例如，如图3C所示，若第一矩阵为第一运算矩阵A，第一矩阵缓存指令中的基地址BaseAddr可以为数据a0的地址，而数据a1的地址可以基于该基地址BaseAddr和数据块A0与数据块A1之间的基地址偏移量计算得到，该数据块A1对应的基地址偏移量可以为数据块A0中一列数据的地址之和，例如，若数据块A0中的一列数据包括16个数据，则该数据块A1对应的基地址偏移量为128字节(双精度)。类似地，若第一矩阵为第二运算矩阵B，第一矩阵缓存指令中的基地址BaseAddr可以为数据b0的地址，而数据b1的地址可以基于该基地址BaseAddr和数据块B0与数据块B1之间的基地址偏移量计算得到，该数据块B1对应的基地址偏移量可以为数据块B0中一行数据的地址之和。

[0183] 需要说明的是，如图6A所示，该数据块A2对应的基地址偏移量可以为数据块A0中一列数据的地址和数据块A1中一列数据的地址之和，例如，若数据块A0中的一列数据包括16个数据，数据块A1中的一列数据包括16个数据，则该数据块A2对应的基地址偏移量为2*128字节(双精度)。

[0184] 例如，如图3C所示，若第一矩阵为第一运算矩阵A，当前线程束为wave2，此时，当前线程束所需的P个数据为第一运算矩阵A中的数据块A1的数据，当前线程束对应的基地址偏移量即为该数据块A1对应的基地址偏移量。如图6A所示，若第一矩阵为第一运算矩阵A，当前线程束为wave2，此时，当前线程束所需的P个数据为第一运算矩阵A中的数据块A2的数据，当前线程束对应的基地址偏移量即为该数据块A2对应的基地址偏移量。

[0185] 例如，对于图3C所示的映射方式，若第一矩阵为第一运算矩阵A，即第一矩阵缓存指令用于实现对第一运算矩阵A的数据进行读取时，在一些示例中，对于wave0～wave3，第一矩阵缓存指令中的BaseAddr可以均为数据a0的地址。wave0和wave1对应的第一读基地址为数据a0的地址，wave0和wave1分别对应的第一矩阵缓存指令中的工作组基地址均为数据a0的地址，从而wave0和wave1对应的第一读基地址可以基于wave0和wave1分别对应的第一矩阵缓存指令直接得到。

[0186] wave2和wave3对应的第一读基地址为数据a1的地址，此时，基于wave2和wave3分别对应的第一矩阵缓存指令，可以得到数据a0的地址(即工作组基地址)，然后，对于wave2，获取wave2对应的基地址偏移量，基于数据a0的地址和wave2对应的基地址偏移量，得到wave2对应的第一读基地址，即数据a1的地址；对于wave3，获取wave3对应的基地址偏移量，基于数据a0的地址和wave3对应的基地址偏移量，得到wave3对应的第一读基地址，即数据

a1的地址。

[0187] 需要说明的是，当第一矩阵为第二运算矩阵B，wave0～wave3对应的第一矩阵缓存指令中的工作组基地址均为数据b0的地址，wave0和wave2对应的第一读基地址为数据b0的地址，wave1和wave3对应的第一读基地址为数据b1的地址，获取各个线程束(wave0～wave3)对应的第一读基地址的过程与上面类似，在此不再赘述。

[0188] 例如，在图3C所示的示例中，对于wave0，当读取第一运算矩阵A的数据时，wave0对应的基地址偏移量为0；当读取第二运算矩阵B的数据时，wave0对应的基地址偏移量为0。对于wave1，当读取第一运算矩阵A的数据时，wave1对应的基地址偏移量为0；当读取第二运算矩阵B的数据时，wave1对应的基地址偏移量为数据块B0中一行数据的地址之和。对于wave2，当读取第一运算矩阵A的数据时，wave2对应的基地址偏移量为数据块A0中一列数据的地址之和；当读取第二运算矩阵B的数据时，wave2对应的基地址偏移量为0。对于wave3，当读取第一运算矩阵A的数据时，wave3对应的基地址偏移量为数据块A0中一列数据的地址之和；当读取第二运算矩阵B的数据时，wave3对应的基地址偏移量为数据块B0中一行数据的地址之和。

[0189] 本公开不限于上述具体实施例，当读取第一运算矩阵A的数据时，wave0～wave3对应的第一读基地址也可以均为数据a0的地址；当读取第二运算矩阵B的数据时，wave0～wave3对应的第一读基地址也可以均为数据b0的地址。

[0190] 例如，在一些实施例中，步骤S30包括：根据第一读基地址和第一排序控制信息，计算得到与第一读基地址对应的多个存储器地址；基于多个存储器地址，从存储器上读取多个待处理数据，其中，多个待处理数据表示存储在存储器的多个存储器地址中的数据；基于第一排序控制信息，从多个待处理数据中获取当前线程束所需的P个数据。

[0191] 例如，在另一些实施例中，步骤S30包括：根据第一读基地址和第一排序控制信息，计算得到与第一读基地址对应的多个存储器地址；基于多个存储器地址，从存储器上读取P个数据，其中，P个数据表示存储在存储器的多个存储器地址中的数据。

[0192] 例如，第一矩阵包括多个第一数据块。在步骤S30中，根据第一读基地址和第一排序控制信息，计算得到与第一读基地址对应的多个存储器地址，包括：根据指令操作码、工作组尺寸和矩阵信息，确定多个第一数据块中与当前线程束对应的至少一个第一数据块，其中，至少一个第一数据块包括待处理数据；根据至少一个第一数据块的数量和运算参数，控制循环状态并输出循环参数；基于至少一个第一数据块中的每个第一数据块的尺寸、循环参数、第一读基地址、地址步长和地址偏移量，计算得到多个存储器地址。

[0193] 例如，对于图3B所示的映射方式，在一些实施例中，若第一矩阵为第二运算矩阵B，当前线程束为wave0，当前线程束对应的至少一个第一数据块为数据块B0和数据块B1，此时，当前线程束所需的P个数据为数据块B0中位于第一列至第四列的数据和数据块B1中位于第一列至第四列的数据。第一读基地址可以为数据块B0的第一行第一列的数据b0的地址，基于该第一读基地址可以计算得到数据块B0和数据块B1中的所有数据在存储器中的地址，即多个存储器地址为数据块B0和数据块B1中的所有数据在存储器中的地址，此时，基于该多个存储器地址即可从存储器上读取得到数据块B0和数据块B1中的所有数据，该数据块B0和数据块B1中的所有数据即为多个待处理数据，然后，基于第一排序控制信息，从多个待处理数据中获取当前线程束所需的P个数据。

[0194] 例如,对于图3B所示的映射方式,在一些实施例中,若第一矩阵为第一运算矩阵A,当前线程束为wave0,当前线程束对应的至少一个第一数据块为数据块A0和数据块A1,此时,当前线程束所需的P个数据为数据块A0和数据块A1的所有数据。第一读基地址可以为数据块A0的第一行第一列的数据a0的地址,基于该第一读基地址可以计算得到数据块A0和数据块A1中的所有数据在存储器中的地址,即多个存储器地址为数据块A0和数据块A1中的所有数据在存储器中的地址,此时,基于该多个存储器地址即可从存储器上读取数据块A0和数据块A1中的所有数据,该数据块A0和数据块A1中的所有数据即为当前线程束所需的P个数据。

[0195] 例如,对于图3C所示的映射方式,在一些实施例中,若第一矩阵为第一运算矩阵A,当前线程束为wave0,当前线程束对应的至少一个第一数据块为数据块A0,此时,当前线程束所需的P个数据为数据块A0的所有数据。第一读基地址可以为数据块A0的第一行第一列的数据a0的地址,基于该第一读基地址可以计算得到数据块A0中的所有数据在存储器中的地址,即多个存储器地址为数据块A0中的所有数据在存储器中的地址,此时,基于该多个存储器地址即可从存储器上读取数据块A0中的所有数据,该数据块A0中的所有数据即为当前线程束所需的P个数据。

[0196] 例如,对于图3C所示的映射方式,在一些实施例中,若第一矩阵为第一运算矩阵A,当前线程束为wave2,当前线程束wave2对应的至少一个第一数据块为数据块A1,此时,当前线程束wave2所需的P个数据为数据块A1中的所有数据。第一读基地址可以为数据块A1的第一行第一列的数据a1的地址,基于该第一读基地址可以计算得到数据块A1中的所有数据在存储器中的地址,即多个存储器地址为数据块A1中的所有数据在存储器中的地址,此时,基于该多个存储器地址即可从存储器上读取得到数据块A1中的所有数据,该数据块A1中的所有数据即为当前线程束wave1所需的P个数据。

[0197] 例如,对于图3C所示的映射方式,在一些实施例中,若第一矩阵为第二运算矩阵B,当前线程束为wave0,当前线程束wave0对应的至少一个第一数据块为数据块B0,此时,当前线程束wave0所需的P个数据为数据块B0中的所有数据。第一读基地址可以为数据块B0的第一行第一列的数据b0的地址,基于该第一读基地址可以计算得到数据块B0中的所有数据在存储器中的地址,即多个存储器地址为数据块B0中的所有数据在存储器中的地址,此时,基于该多个存储器地址即可从存储器上读取得到数据块B0中的所有数据,该数据块B0中的所有数据即为当前线程束wave0所需的P个数据。

[0198] 例如,对于图3C所示的映射方式,在一些实施例中,若第一矩阵为第二运算矩阵B,当前线程束为wave1,当前线程束wave1对应的至少一个第一数据块为数据块B1,此时,当前线程束wave1所需的P个数据为数据块B1中的所有数据。第一读基地址可以为数据块B1的第一行第一列的数据b1的地址,基于该第一读基地址可以计算得到数据块B1中的所有数据在存储器中的地址,即多个存储器地址为数据块B1中的所有数据在存储器中的地址,此时,基于该多个存储器地址即可从存储器上读取得到数据块B1中的所有数据,该数据块B1中的所有数据即为当前线程束wave1所需的P个数据。

[0199] 例如,在一些实施例中,基于至少一个第一数据块中的每个第一数据块的尺寸、循环参数、第一读基地址、地址步长和地址偏移量,计算得到多个存储器地址,包括:基于至少一个第一数据块中的每个第一数据块的尺寸、循环参数、第一读基地址、地址步长和地址偏

移量,计算得到多个首地址;基于多个首地址,计算多个存储器地址。

[0200] 例如,至少一个第一数据块中的每个第一数据块的数据排列为多行多列,在第一矩阵为第一运算矩阵A的情况下,多个首地址为至少一个第一数据块的各列数据中的第一个数据的地址,也就是说,多个首地址为每个第一数据块的位于第一行的数据的地址;在第一矩阵为第二运算矩阵B的情况下,多个首地址为至少一个第一数据块的各行数据的第一个数据的地址,也就是说,多个首地址为每个第一数据块的位于第一列的数据的地址。

[0201] 在基于MMBUF进行地址计算的过程中,在获取基地址BaseAddr以后,即可以由地址计算子电路(硬件电路)根据矩阵的形状算出每个数据块中所有数据在存储器中的地址。

[0202] 例如,如果第一运算矩阵A是列主序类型的矩阵,则每个尺寸为16*4数据块中的位于同一列的数据,即16*1的数据,对应的地址(在存储器中的地址)是连续的,每相邻两列数据(即两个16*1的数据)之间,在地址上的距离为一个“STRIDE(存储在MMBUF指令中)”,“STRIDE”的具体数值一般由应用程序指定并作为内核(Kernel)的参数保存在硬件的寄存器里。在本公开的实施例中,以上的信息由硬件调度电路传递给读取与排序电路中的地址计算子电路,地址计算子电路根据Opcode、WGSHAPE、K解码出:在工作组中在各方向上线程束的个数;每个线程束对应第一运算矩阵A中的数据块的个数和数据块的尺寸和第二运算矩阵B中的数据块的个数和数据块的尺寸。

[0203] 例如,如果采用X表示在行方向(例如,水平方向)上工作组中的每一行线程束的个数,采用Y表示在列方向(例如,竖直方向)上工作组中的每一列线程束的个数,若WGSHAPE=16*16,则X=2,Y=2;若WGSHAPE=8*32,则X=4,Y=1,等等。Opcode决定了每个线程束对应的第一运算矩阵A中的数据块的个数和对应的第二运算矩阵B中的数据块的个数,例如,对于Opcode:mmbuf_load2*2_dword,其中,2*2表示:在竖直方向上,每个线程束对应的第一运算矩阵A中的2个数据块,在水平方向上,每个线程束对应的第二运算矩阵B中的2个数据块。

[0204] 例如,在图3C所示的示例中,每个线程束(wave0/wave1/wave2/wave3)需要读取第一运算矩阵A的一个数据块(尺寸为16*4)和第二运算矩阵B的一个数据块(尺寸为4*16);在图6A所示的示例中,每个线程束需要读取第一运算矩阵A的两个数据块(每个数据块的尺寸为16*4)和第二运算矩阵B的两个数据块(每个数据块的尺寸为4*16);在图6B所示的示例中,每个线程束需要读取第一运算矩阵A的三个数据块(每个数据块的尺寸为16*4)和第二运算矩阵B的三个数据块(每个数据块的尺寸为4*16);在图6C所示的示例中,每个线程束需要读取第一运算矩阵A的四个数据块(每个数据块的尺寸为16*4)和第二运算矩阵B的四个数据块(每个数据块的尺寸为4*16)。

[0205] K值决定了数据块的尺寸,若K=4则代表数据块的尺寸为16*4或4*16,若K=8则代表数据块的尺寸为16*8或8*16,一般情况下K为4的倍数。

[0206] 例如,硬件调度电路可以解析MMBUF中的V#,以得到每个线程束对应的第一运算矩阵A或第二运算矩阵B的BaseAddr,然后将该BaseAddr发送给地址计算子电路,地址计算子电路根据指令取数据的粒度算出每个数据的地址。假如,对于双精度(一个数据是8个字节),在存储器上,第一运算矩阵A的第一个数据块(尺寸为16*4)的数据对应的地址如下表2所示:

[0207] 表2

[0208]

16*1	BaseAddr + 0*STRIDE +															
0	0	8	16	24	32	40	48	56	64	72	80	88	96	104	112	120
	BaseAddr + 1*STRIDE +															
1	0	8	16	24	32	40	48	56	64	72	80	88	96	104	112	120
	BaseAddr + 2*STRIDE +															
2	0	8	16	24	32	40	48	56	64	72	80	88	96	104	112	120
	BaseAddr + 3*STRIDE +															
3	0	8	16	24	32	40	48	56	64	72	80	88	96	104	112	120

[0209] 如上面表2所示，第一列的0～3分别表示第一个数据块的四列数据的编号，0表示第一列数据，1表示第二列数据，以此类推。每列数据具有16*1个数据。第一行(编号为0)表示第一个数据块的第一列数据的地址，第一行的第一个地址为BaseAddr+0*STRIDE+0，即第一列数据的第一个数据的地址，第一行的第二个地址为BaseAddr+0*STRIDE+8，即第一列数据的第二个数据的地址，依次类推。

[0210] 若线程束对应有第一运算矩阵A的第二个数据块，则在存储器上，第二个数据块的数据对应的地址如下表3所示：

[0211] 表3

[0212]

16*1	BaseAddr + 0*STRIDE +128															
0	0	8	16	24	32	40	48	56	64	72	80	88	96	104	112	120
	BaseAddr +1* STRIDE +128															
1	0	8	16	24	32	40	48	56	64	72	80	88	96	104	112	120
	BaseAddr + 2*STRIDE +128															
2	0	8	16	24	32	40	48	56	64	72	80	88	96	104	112	120
	BaseAddr + 3*STRIDE +128															
3	0	8	16	24	32	40	48	56	64	72	80	88	96	104	112	120

[0213] 如上面表3所示，“BaseAddr+x(0、1、2、3)*STRIDE+128”中的128表示第二个数据块相对于第一个数据块的地址偏移量。若还有更多的数据块则以此类推。

[0214] 一个数据块在数据总线上的传输时间取决于这个线程束对应这个数据块的数量、数据粒度的大小(dword或者dword*2)以及数据总线的宽度。例如，在一些实施例中，地址计算子电路、数据排序子电路以及读取与缓存子电路的数据总线宽度均为512比特，由于同一列16*1数据的地址是连续的，所以根据连续地址可以合并的原理，对于一个数据块，地址计算子电路在进行地址计算时实际上只需计算各列数据的第一个数据的地址即可，例如，对于第一运算矩阵A的数据块(尺寸为16*4)，则地址计算子电路在进行地址计算时实际上只需计算该数据块的每列数据的第一个数据(对于四列数据，即需要计算四个数据)的地址即可；对于第二运算矩阵B的数据块，则地址计算子电路在进行地址计算时实际上只需计算该数据块的每行数据的第一个数据(对于四行数据，即需要计算四个数据)的地址即可。

[0215] 例如,STRIDE和BaseAddr来自于MMBUF指令封装的信息。对于列主序类型的第一运算矩阵A的数据块,若第一运算矩阵A的数据块的一列数据包括16个数据,在单精度下,地址偏移量是64字节,在双精度下,地址偏移量是128字节。

[0216] 例如,BaseAddr表示第一个数据块的第一列的第一个数据的地址,第一运算矩阵A的第一个数据块的第一列的第一个数据相对于BaseAddr的地址偏移量是0字节,第一运算矩阵A的第二个数据块的第一列的第一个数据相对于BaseAddr的地址偏移量为128字节(双精度),第一运算矩阵A的第三个数据块的第一列的第一个数据相对于第一运算矩阵A的第二个数据块的第一列的第一个数据的地址偏移量为128字节(双精度)。

[0217] 例如,对于列主序类型的第一运算矩阵A的数据块,在存储器中,每个数据块的相邻两列数据之间的地址步长用STRIDE表示,例如,数据块的第二列和第一列之间的步长为1个STRIDE,数据块的第三列与第二列之间的步长为1个STRIDE,数据块的第三列与第一列之间的步长为2个STRIDE。每列数据的基地址的公式为BaseAddr+STRIDE*j,其中,j表示当前数据列的列数,对于第一列数据,j为1,对于第二列数据,j为2,等等。行主序的第二运算矩阵B的数据块的地址计算过程与列主序的第一运算矩阵A的数据块的地址计算过程类似,不再赘述。

[0218] 例如,在一些实施例中,步骤S40包括:根据第一写基地址和第一排序控制信息,计算得到与P个数据对应的多个寄存器地址,其中,多个寄存器地址为多个矢量通用寄存器片上用于写入P个数据的地址,多个寄存器地址表示执行矩阵乘法运算时P个数据对应的存储地址;基于第一排序控制信息,确定P个数据与多个寄存器地址的对应关系,基于对应关系,将P个数据写入多个寄存器地址中。

[0219] 例如,根据第一写基地址和第一排序控制信息,计算得到与P个数据对应的多个寄存器地址,包括:根据第一写基地址、指令操作码、工作组尺寸和运算参数,计算得到多个寄存器地址。

[0220] 例如,在将矩阵的数据从存储器读回来后需要针对矩阵乘法运算时数据在矢量通用寄存器里的排列方式进行重排,在进行数据重排时,需要计算数据对应在矢量通用寄存器中的地址,即上述多个寄存器地址。数据重排的方式依赖于进行矩阵乘法运算时线程束的映射方式。

[0221] 例如,以图6A所示的映射方式为例进行数据重排的说明,图7A为图6A示例中单个线程束的映射方式,因为矢量通用寄存器通常以32比特为单位,为了方便描述,图7A以单精度运算为例进行第一运算矩阵A和第二运算矩阵B中的每个数据对应矢量通用寄存器的索引地址(即寄存器地址)的说明。需要说明的是,在图7A和7B所示的示例中,子矩阵块C00由一个线程束执行。

[0222] 图7A示出了第一运算矩阵A和第二运算矩阵B中的每个数据对应的寄存器地址,即矢量通用寄存器的索引地址,例如,第一运算矩阵A包括数据块A0和数据块A1,数据块A0和数据块A1的尺寸均为16*4,第二运算矩阵B包括数据块B0和数据块B1,数据块B0和数据块B1的尺寸均为4*16。

[0223] 第一运算矩阵A的数据块A0和数据块A1对应的寄存器地址为V0至V7,例如,数据块A0中的第一列数据对应的寄存器地址均为v0,例如,ROW0的v0;数据块A0中的第二列数据对应的寄存器地址均为v1,例如,ROW1的v1;数据块A0中的第三列数据对应的寄存器地址均为

v2,例如,ROW2的v2;数据块A0中的第四列数据对应的寄存器地址均为v3,例如,ROW3的v3;数据块A1中的第一列数据对应的寄存器地址均为v4,例如,ROW0的v4;数据块A1中的第二列数据对应的寄存器地址均为v5,例如,ROW1的v5;数据块A1中的第三列数据对应的寄存器地址均为v6,例如,ROW2的v6;数据块A1中的第四列数据对应的寄存器地址均为v7,例如,ROW3的v7。

[0224] 第二运算矩阵B的数据块B0和数据块B1对应的寄存器地址为V8至V39,例如,数据块B0中的第一行的第一个数据至第四个数据对应的寄存器地址均为v8;数据块B0中的第二行的第一个数据至第四个数据对应的寄存器地址均为v9;数据块B0中的第三行的第一个数据至第四个数据对应的寄存器地址均为v10,数据块B0中的第四行的第一个数据至第四个数据对应的寄存器地址均为v11,依次类推。

[0225] 例如,如图7A所示,虽然数据块B0中的第一行的第一个数据至第四个数据对应的寄存器地址均为v8,然而,如图8A～8D所示,数据块B0中的第一行的第一个数据B00对应的寄存器地址为ROW0的v8,数据块B0中的第一行的第二个数据B01对应的寄存器地址为ROW1的v8,数据块B0中的第一行的第三个数据B02对应的寄存器地址为ROW2的v8,数据块B0中的第一行的第四个数据B03对应的寄存器地址为ROW3的v8。其余数据与此类似,不再赘述。

[0226] 图7B示出了第一运算矩阵A和第二运算矩阵B中的各个数据。例如,如图7B所示,数据块A0包括数据A00、A01、A02、A03、A10、…、A150、A151、A152、A153,数据块A1包括数据A160、A161、A162、A163、…、A310、A311、A312、A313,数据块B0包括数据B00、B01、B02、B03、…、B015、B115、B215、B315,数据块B1包括数据B016、B116、B216、B316、…、B031、B131、B231、B331。

[0227] 例如,在一些实施例中,在步骤S40中,基于对应关系,将P个数据写入多个寄存器地址中,包括:响应于第一矩阵为第一运算矩阵:基于对应关系,直接将P个数据写入多个寄存器地址中。

[0228] 例如,响应于第一矩阵为第一运算矩阵,基于对应关系,直接将P个数据写入多个寄存器地址中,包括:将P个数据被划分为与多个矢量通用寄存器片一一对应的多个数据组;直接将多个数据组分别写入多个矢量通用寄存器片上。

[0229] 例如,在矩阵乘法运算中,第一运算矩阵A的数据不需要进行复制,可以直接被存储至对应的寄存器地址中。在一些实施例中,每个单指令多数据流包括四个矢量通用寄存器片ROW0～ROW3,从而P个数据可以被划分为四个数据组。若P个数据为第一运算矩阵A的一个数据块(尺寸为16*4)中的数据,此时,可以将第一运算矩阵A的一个数据块的四列数据分别作为四个数据组,即第一运算矩阵A的一个数据块的四列数据一一对应地存储在四个矢量通用寄存器片ROW0～ROW3。此时,由于每列数据仅写入一个矢量通用寄存器片中,而不必写入多个矢量通用寄存器片,从而可以减少将第一运算矩阵A的数据块写入矢量通用寄存器片上的时间。

[0230] 由于矩阵乘法运算时,第一运算矩阵A的数据块的四列数据在每个矢量通用寄存器片都会用到,这里可以利用DPPM指令特性从不同的矢量通用寄存器片上取第一运算矩阵A的数据。

[0231] 例如,按照数据总线宽度为512比特来算,在单精度的矩阵乘法运算中,将第一运算矩阵A的一个数据块(尺寸为16*4)写入矢量通用寄存器需要4个时钟周期。

[0232] 例如,在另一些实施例中,第一运算矩阵A的一个数据块的每列数据也可以分别写入四个矢量通用寄存器片ROW0～ROW3,即每个矢量通用寄存器片均存储第一运算矩阵A的一个数据块的所有数据。

[0233] 例如,在一些实施例中,在步骤S40中,基于对应关系,将P个数据写入多个寄存器地址中包括:响应于第一矩阵为第二运算矩阵:获取P个数据中的第a个数据,其中,a为正整数;将第a个数据复制D份,其中,D为大于1的正整数;基于对应关系,确定多个寄存器地址中与第a个数据对应的寄存器地址;将D份第a个数据写入与第a个数据对应的寄存器地址中。

[0234] 例如,从第二运算矩阵B的数据在矢量通用寄存器中的排列方式来看,第二运算矩阵B中的每个数据在写入矢量通用寄存器之前需要进行复制,例如,D可以为16,此时,数据B00需要被复制16份,然后被写入ROW0上,数据B01需要被复制16份,然后被写入ROW1上,数据B02需要被复制16份,然后被写入ROW2上,数据B03需要被复制16份,然后被写入ROW3上,以此类推。

[0235] 按照数据总线宽度为512比特来算,在单精度的矩阵乘法运算中,将第二运算矩阵B的一个数据块(尺寸为4*16)写进矢量通用寄存器需要64个时钟周期,由于一个数据块的读请求发送时间是四个时钟周期,所以如果是连续读取多个数据块(尺寸均为4*16)的数据,则除了多个数据块中的第一个数据块,其他数据块要在读回的路径上多等待60个时钟周期。为了解决这个问题,可以在将第二运算矩阵B的数据写入矢量通用寄存器前,将第二运算矩阵B的数据通过一个数据缓冲子电路进行缓存,将已经读回来的第二运算矩阵B的数据块先保存在数据缓冲子电路中。由于ALU寄存器硬件限制的原因,一个工作组一次能计算的最大矩阵是双精度的128*128的矩阵,按照一个工作组具有4个线程束,每个线程束计算64*64的双精度的矩阵,那么最大需要的数据缓冲子电路中的缓冲数据量为64*4*8＝2KB。例如,该数据缓冲子电路可以设置在数据读回和写进矢量通用寄存器之间的通路上。

[0236] 需要说明的是,D的具体数值可以基于矩阵乘法运算确定。

[0237] 图8A～8D示出了图7B所示的数据块A0、数据块A1、数据块B0和数据块B1在矢量通用寄存器片中的数据排列方式。图8A示出了ROW0中的数据排列方式,图8B示出了ROW1中的数据排列方式,图8C示出了ROW2中的数据排列方式,图8D示出了ROW3中的数据排列方式。

[0238] 需要说明的是,在图8A～8D中,A0,0即表示图7B中的A00,B0,0即表示图7B中的B00,以此类推。每个矢量通用寄存器片的一行表示一个矢量通用寄存器,例如,矢量通用寄存器片ROW0中的v0表示一个矢量通用寄存器,以此类推。

[0239] 假设一个工作组计算的是单精度的(64*4)*(4*64)的矩阵,一个工作组包括四个线程束wave0～wave3。

[0240] 对于单精度的矩阵(64*64＝(64*4)*(4*64))的矩阵乘法运算,基于图6A所示的映射方式,该矩阵乘法运算可以通过四个线程束实现,此时,一个线程束用于实现矩阵的一个子矩阵块(即32*32＝(32*4)*(4*32))的矩阵乘法运算,此时,第一运算矩阵的尺寸为32*4,第二运算矩阵的尺寸为4*32,第一运算矩阵被划分为两个数据块A0和A1,第二运算矩阵被划分为两个数据块B0和B1。例如,线程束wave0读取第一运算矩阵A的数据块A0(16*4)和数据块A1(16*4),以及第二运算矩阵B的数据块B0(4*16)和数据块B1(4*16);线程束wave1读取第一运算矩阵A的数据块A0和的数据块A1,以及第二运算矩阵B的数据块B2和数据块B3;线程束wave2读取第一运算矩阵A的数据块A2和数据块A3,以及第二运算矩阵B的数据块B0

和数据块B1;线程束wave3读取第一运算矩阵A的数据块A2和数据块A3,以及第二运算矩阵B的数据块B2和数据块B3。

[0241] 对于采用现有的数据读取方式,一个线程束读取数据的总时间延迟的计算过程如下所述:

[0242] 1、将第一运算矩阵A(尺寸为64*4)的数据和第二运算矩阵B(尺寸为4*64)的数据从存储器读回并写入到矢量通用寄存器的延迟时间为:(64*4Byte(对于单精度数据,若采用双精度数据,则此处为8Byte)*4*2)/64Byte=32cycles(时钟周期)。

[0243] 2、片上存储器的读写端口的宽度是128Byte,所以将第一运算矩阵A和第二运算矩阵B的数据再从矢量通用寄存器中写入片上存储器需要16cycles。

[0244] 3、将第一运算矩阵A和第二运算矩阵B的数据从片上存储器读出并按照矩阵乘法运算的排序方式写入矢量通用寄存器。在理想情况下,一个线程束读取第一运算矩阵A的一个尺寸为16*4的数据块需要4cycles,由于不同的单指令多数据流访问片上存储器时,片上存储器对每个单指令多数据流的响应是要排队的,所以在最差的情况下,4个单指令多数据流同时读取片上存储器时,一共需要16cycles读取第一运算矩阵A的一个尺寸为16*4的数据块,读取第一运算矩阵A的两个数据块需要32cycles。对于第二运算矩阵B的数据块,根据上述描述,每个cycle只读第二运算矩阵B的一个数据,一个线程束读取第二运算矩阵B的一个尺寸为4*16的数据块需要64cycles,读取第二运算矩阵B的两个数据块共需要128cycles,在最差的情况下,四个单指令多数据流排队读取片上存储器则需要512cycle才能完成数据传输。

[0245] 由此可知,在最理想情况下,对于一个线程束,最短的延迟时间是32+16+4+128=180cycles,最长的延迟时间是32+16+16+512=576cycles。四个单指令多数据流不一定会同时读取片上存储器,假如四个单指令多数据流读取片上存储器的时间是完全不重叠的,那么对于每个单指令多数据流来说完成整个数据准备的时间是180cycles,其中,读取片上存储器的时间是:4+128cycles;假如四个单指令多数据流读取片上存储器的时间是完全重叠的,那么四个单指令多数据流必须排队读取片上存储器,这样势必会有一个单指令多数据流排队的时间最长,该最长的读取片上存储器的时间就是四个单指令多数据流的时间和576:16+512=4*(4+128)cycles。

[0246] 在本公开的实施例提供的数据读取方法中,在利用MMBUF进行数据读取时,对于一个线程束,将数据写入矢量通用寄存器的时间延迟的计算过程如下所述:

[0247] 对于第一运算矩阵A:(16*4*4(B:byte,字节)*2(数据块))/64B=8cycles;

[0248] 对于第二运算矩阵B:16*4*2(数据块)=128cycles。

[0249] 一个线程束的总时间延迟为:128+8=136cycles,一个工作组的总时间延迟是136*4=544cycles。

[0250] 在进行矩阵乘法运算时,分别计算A0*B0、A0*B1、A1*B0和A1*B1即可实现该矩阵乘法运算,A0*B0、A0*B1、A1*B0和A1*B1中的每个进行运算时所需的时钟周期是64cycles,那么对于这个线程束而言,完成矩阵乘法运算一共需要256cycles。由此可见,在本公开的实施例中,矩阵乘法运算的时间比数据读取的时间要长,可以利用矩阵乘法运算的延迟来弥补上述读取数据上的延迟。

[0251] 综上所述,本公开的实施例提供的数据读取方法进行数据读取时,时间延迟较小,

对于现有的数据读取方式，还需要考虑片上存储器内部流水线的延迟，由于数据在片上存储器内部RAM地址上的冲突(Bank Conflict)导致的额外延迟的增加；还需要考虑到数据地址计算产生的指令开销，矢量通用寄存器和片上存储器之间相互搬运数据产生的指令开销，以及相应产生的功耗。因此，基于本公开的实施例提供的MMBUF指令进行数据读取具备性能和功耗方面的优势，减小数据读取总延迟，节省功耗，提高矩阵运算效率。

[0252] 本公开一些实施例还提供一种数据读取电路。图9为本公开一些实施例提供的一种数据读取电路的示意图。

[0253] 例如，数据读取电路可以应用于矩阵乘法运算，例如，该数据读取电路可以用于实现上述实施例所述的数据读取方法。

[0254] 例如，如图9所示，数据读取电路900可以包括硬件调度电路901和读取与排序电路902，硬件调度电路901和读取与排序电路902耦接，硬件调度电路901和读取与排序电路902之间可以实现数据传输。

[0255] 例如，硬件调度电路901被配置为：获取与当前线程束对应的第一矩阵缓存指令，其中，第一矩阵缓存指令用于实现对第一矩阵的数据进行读取；解析第一矩阵缓存指令，以获得与当前线程束所需的第一矩阵中的P个数据对应的第一读基地址、第一排序控制信息和第一写基地址，其中，P为正整数。也就是说，硬件调度电路901可以实现图4所示的步骤S10和S20。

[0256] 例如，在一些实施例中，在执行获取与当前线程束对应的第一矩阵缓存指令的步骤时，硬件调度电路被配置为：接收与当前线程束对应的命令；基于命令，读取第一矩阵缓存指令。

[0257] 例如，如图9所示，与当前线程束对应的命令可以由交互接口电路910(即SPI：Software-Program-Interface)发出。

[0258] 例如，在一些实施例中，在执行解析第一矩阵缓存指令，以获得第一读基地址的步骤时，硬件调度电路901被配置为：解析第一矩阵缓存指令，以得到当前线程束所在的工作组对应的工作组基地址；获取当前线程束对应的基地址偏移量；基于工作组基地址和基地址偏移量，得到第一读基地址。

[0259] 例如，在另一些实施例中，在执行解析第一矩阵缓存指令，以获得第一读基地址的步骤时，硬件调度电路901被配置为：解析第一矩阵缓存指令，以得到当前线程束所在的工作组对应的工作组基地址；将工作组基地址作为第一读基地址。

[0260] 例如，读取与排序电路902被配置为：基于第一读基地址和第一排序控制信息，从存储器上读取P个数据；基于第一写基地址和第一排序控制信息，按照执行矩阵乘法运算时P个数据对应的存储地址将P个数据写入与当前线程束对应的单指令多数据流单元的多个矢量通用寄存器片上。也就是说，读取与排序电路902可以实现图4所示的步骤S30和S40。

[0261] 例如，在一些实施例中，在执行解析第一矩阵缓存指令，以获得第一排序控制信息的步骤时，硬件调度电路901被配置为：解析第一矩阵缓存指令，以得到封装在第一矩阵缓存指令中的指令操作码、工作组尺寸、地址偏移量、地址步长、矩阵信息和运算参数。

[0262] 例如，第一排序控制信息包括指令操作码、工作组尺寸、地址偏移量、地址步长、矩阵信息和运算参数。

[0263] 例如，矩阵乘法运算用于实现将第一运算矩阵和第二运算矩阵进行乘法计算，若

第一运算矩阵表示为M*K,第二运算矩阵表示为K*N,则运算参数为K,M、K、N均为正整数。

[0264] 例如,矩阵信息用于指示第一矩阵为第一运算矩阵或第二运算矩阵以及第一矩阵是否被转置,在第一矩阵为第一运算矩阵(列主序类型的矩阵)的情况下,地址步长表示第一矩阵的相邻两列数据之间的步长,在第一矩阵为第二运算矩阵(行主序类型的矩阵)的情况下,地址步长表示第一矩阵的相邻两行数据之间的步长。

[0265] 例如,如图9所示,读取与排序电路902包括地址计算子电路9021(TA)、数据排序子电路9022(TD)和读取与缓存子电路9023。地址计算子电路9021和数据排序子电路9022耦接,地址计算子电路9021和读取与缓存子电路9023耦接,数据排序子电路9022和读取与缓存子电路9023耦接。

[0266] 例如,地址计算子电路9021被配置为:根据第一读基地址和第一排序控制信息,计算得到与第一读基地址对应的多个存储器地址。

[0267] 在一些实施例中,读取与缓存子电路9023被配置为:基于多个存储器地址,从存储器上读取多个待处理数据;并将多个待处理数据进行缓存。多个待处理数据表示存储在存储器的多个存储器地址中的数据。数据排序子电路9022被配置为:获取多个待处理数据;基于第一排序控制信息,从多个待处理数据中获取当前线程束所需的P个数据。例如,数据排序子电路9022可以从读取与缓存子电路9023中读取该多个待处理数据,并进行后续处理。

[0268] 在另一些实施例中,读取与缓存子电路9023被配置为:基于多个存储器地址,从存储器上读取P个数据;并将P个数据进行缓存。P个数据表示存储在存储器的多个存储器地址中的数据。数据排序子电路9022被配置为:直接从读取与缓存子电路9023上读取该P个数据,以进行后续处理。

[0269] 图10为本公开一些实施例提供的一种地址计算子电路的结构示意图。

[0270] 例如,如图10所示,地址计算子电路9021包括解码模块110、循环模块120和第一计算模块130。

[0271] 例如,在一些实施例中,第一矩阵包括多个第一数据块。

[0272] 例如,解码模块110被配置为:根据指令操作码、工作组尺寸和矩阵信息,确定多个第一数据块中与当前线程束对应的至少一个第一数据块。至少一个第一数据块包括多个待处理数据。

[0273] 例如,解码模块110可以根据封装在第一矩阵缓存指令中的Opcode、WGSHAPE和AB(即表示该第一矩阵缓存指令对应第一运算矩阵A还是第二运算矩阵B),计算得到与当前线程束对应的至少一个第一数据块。

[0274] 例如,每个第一数据块的尺寸为16*K,即包括16*K个数据,由于地址总线带宽的限制,一个时钟周期,地址计算子电路9021只能对一个第一数据块中的一列的数据(即16个数据)发送读取请求。如图3B和图3C所示,当第一矩阵为第一运算矩阵A时,至少一个第一数据块包括数据块A0和数据块A1;当第一矩阵为第二运算矩阵B时,至少一个第一数据块包括数据块B0和数据块B1。

[0275] 例如,对于图3B和图3C所示的数据块A0和数据块A1,读取数据的过程为:在时钟周期0,发送数据块A0的第一列数据的读取请求,在时钟周期1,发送数据块A0的第二列数据的读取请求,一直到时钟周期K-1,发送数据块A0的第K列数据的读取请求,从而完成数据块A0的读取请求发送;然后,在时钟周期K,发送数据块A1的第一列数据的读取请求,在时钟周期

K+1,发送数据块A1的第二列数据的读取请求,一直到时钟周期2K-1,发送数据块A1的第K列数据的读取请求,从而完成数据块A1的读取请求发送。

[0276] 例如,如图10所示,循环模块120被配置为:根据至少一个第一数据块的数量和运算参数K,控制循环状态并输出循环参数。

[0277] 例如,在每次时钟周期中,读取与缓存子电路9023只能利用线程读取一个第一数据块的一列/行数据(当第一数据块为数据块A0或A1时,读取一列数据;当第一数据块为数据块B0或B1时,读取一行数据),一列/行数据包括16个数据。例如,循环模块120可以实现以下功能:

[0278] {for(i=0;i<数据块的数量;i++)

[0279] {

[0280] for(j=0;j<K;j++)//每个数据块有K列/行,每次时钟周期读取一列/行的数据

[0281] }

[0282] }

[0283] 例如,循环参数包括第一参数i和第二参数j,第一参数i表示当前需要计算地址的数据所在的数据块的编号,例如,对于第一运算矩阵A,数据块A0的编号为0,数据块A1的编号为1;对于第二运算矩阵B,数据块B0的编号为0,数据块B1的编号为1。第二参数j表示当前需要计算地址的数据所在的行/列的行/列数。

[0284] 例如,第一计算模块130被配置为:基于至少一个第一数据块中的每个第一数据块的尺寸、循环参数、第一读基地址、地址步长和地址偏移量,计算得到多个存储器地址。

[0285] 例如,第一计算模块130用于计算第一数据块的一列数据(当存储器存储第一数据块时,按列进行存储,即在存储器中,该第一数据块的位于同一列的数据的地址是连续的)或一行数据(当存储器存储第一数据块时,按行进行存储,即在存储器中,该第一数据块的位于同一行的数据的地址是连续的)的首地址,即第一数据块的一列数据或一行数据的第一个数据的地址。

[0286] 例如,如图10所示,在一些实施例中,第一计算模块130包括第一计算子模块1301和第二计算子模块1302。

[0287] 例如,第一计算子模块1301被配置为:基于至少一个第一数据块中的每个第一数据块的尺寸、循环参数、第一读基地址、地址步长和地址偏移量,计算得到多个首地址。例如,至少一个第一数据块中的每个第一数据块的数据排列为多行多列,在第一矩阵为第一运算矩阵的情况下,多个首地址为至少一个第一数据块的各列数据中的第一个数据的地址,在第一矩阵为第二运算矩阵的情况下,多个首地址为至少一个第一数据块的各行数据的第一个数据的地址。

[0288] 例如,如图10所示,第一计算子模块1301包括乘法器、加法器和触发器,乘法器用于将第一参数i和偏移量相乘以得到第一乘法结果,即偏移量*i,还用于将第二参数j和STRIDE相乘以得到第二乘法结果,即STRIDE*j;加法器用于将第一乘法结果、第二乘法结果和BaseAddr进行相加,从而得到第一数据块中的一列数据或一行数据在存储器中的首地址;触发器用于存储该首地址,并基于时钟信号的控制,将该首地址发送至第二计算子模块1302。

[0289] 例如,第二计算子模块1302被配置为:基于多个首地址,计算多个存储器地址。

[0290] 例如，如图10所示，第二计算子模块1302用于根据第一数据块中的一列数据或一行数据在存储器中的首地址计算这一列数据或者一行数据里所有数据的地址，因为这一列数据或者一行数据在存储器中的地址是连续的，所以根据每个第一数据的大小(双精度是8字节，单精度是4字节)，计算出一列数据或一行数据中的所有数据的地址。例如，第二计算子模块1302中的各个矩形格示出的“+0、+8、+16、…、+120”，“+0”表示将计算出的一列数据或一行数据的首地址加上0以得到一列数据或一行数据中的第一个数据，“+8”表示将计算出的一列数据或一行数据的首地址加上8以得到一列数据或一行数据中的第二个数据，依次类推，“+120”表示将计算出的一列数据或一行数据的首地址加上120以得到一列数据或一行数据中的最后一个数据，以此类推。

[0291] 例如，在一些实施例中，数据排序子电路9022还被配置为：根据第一写基地址和第一排序控制信息，计算得到与P个数据对应的多个寄存器地址；基于第一排序控制信息，确定P个数据与多个寄存器地址的对应关系，基于对应关系，将P个数据写入多个寄存器地址中。

[0292] 例如，多个寄存器地址为多个矢量通用寄存器片上用于写入P个数据的地址，多个寄存器地址表示执行矩阵乘法运算时P个数据对应的存储地址。

[0293] 例如，在执行根据第一写基地址和第一排序控制信息，计算得到与P个数据对应的多个寄存器地址的步骤时，数据排序子电路9022被配置为：根据第一写基地址、指令操作码、工作组尺寸和运算参数，计算得到多个寄存器地址。

[0294] 需要说明的是，在另一些实施例中，多个寄存器地址也可以由地址计算子电路9021计算得到，然后地址计算子电路9021将该多个寄存器地址传输至数据排序子电路9022，以供数据排序子电路9022使用。

[0295] 例如，在执行基于对应关系，将P个数据写入多个寄存器地址中的步骤时，数据排序子电路9022被配置为：响应于第一矩阵为第一运算矩阵：基于对应关系，直接将P个数据写入多个寄存器地址中；响应于第一矩阵为第二运算矩阵：获取P个数据中的第a个数据，其中，a为正整数；将第a个数据复制D份，其中，D为大于1的正整数；基于对应关系，确定多个寄存器地址中与第a个数据对应的寄存器地址；将D份第a个数据写入与第a个数据对应的寄存器地址中。

[0296] 例如，在执行响应于第一矩阵为第一运算矩阵，基于对应关系，直接将P个数据写入多个寄存器地址中的步骤时，数据排序子电路9022被配置为：将P个数据被划分为与多个矢量通用寄存器片一一对应的多个数据组；直接将多个数据组分别写入多个矢量通用寄存器片上。

[0297] 例如，在一些实施例中，硬件调度电路901还被配置为：获取与当前线程束对应的第二矩阵缓存指令，其中，第二矩阵缓存指令用于实现对第二矩阵的数据进行读取；解析第二矩阵缓存指令，以获得与当前线程束所需的第二矩阵中的Q个数据对应的第二读基地址、第二排序控制信息和第二写基地址，其中，Q为正整数。

[0298] 例如，读取与排序电路902还被配置为：基于第二读基地址和第二排序控制信息，从存储器上读取Q个数据；基于第二写基地址和第二排序控制信息，按照执行矩阵乘法运算时Q个数据对应的存储地址将Q个数据写入单指令多数据流单元的多个矢量通用寄存器片。

[0299] 例如，当前线程束使用写入单指令多数据流单元的多个矢量通用寄存器片上的P

个数据和Q个数据进行矩阵乘法运算。

[0300] 例如，如图9所示，硬件调度电路901和读取与排序电路902还与计算单元CU耦接，以实现数据传输。计算单元CU包括四个单指令多数据流和片上存储器，每个单指令多数据流包括ALU和一组矢量通用寄存器。每组矢量通用寄存器包括四个矢量通用寄存器片ROW0～ROW3。

[0301] 例如，硬件调度电路901和读取与排序电路902可以通过寄存器、触发器、逻辑门电路等硬件实现。

[0302] 需要说明的是，对于数据读取电路的相关说明可以参考上述对数据读取方法的实施例的内容，重复之处不再赘述。

[0303] 对于本公开，还有以下几点需要说明：

[0304] (1)本公开实施例附图只涉及到与本公开实施例涉及到的结构，其他结构可参考通常设计。

[0305] (2)为了清晰起见，在用于描述本发明的实施例的附图中，层或结构的厚度和尺寸被放大。可以理解，当诸如层、膜、区域或基板之类的元件被称作位于另一元件“上”或“下”时，该元件可以“直接”位于另一元件“上”或“下”，或者可以存在中间元件。

[0306] (3)在不冲突的情况下，本公开的实施例及实施例中的特征可以相互组合以得到新的实施例。

[0307] 以上所述仅为本公开的具体实施方式，但本公开的保护范围并不局限于此，本公开的保护范围应以所述权利要求的保护范围为准。

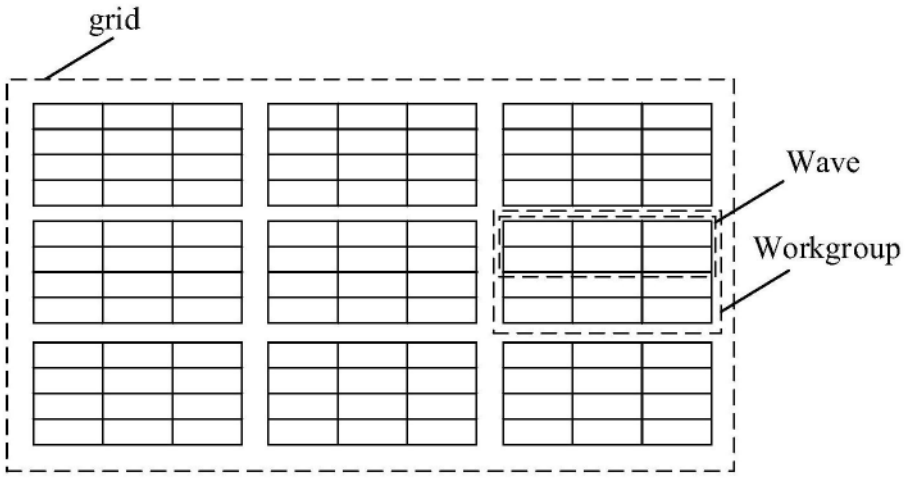


图1

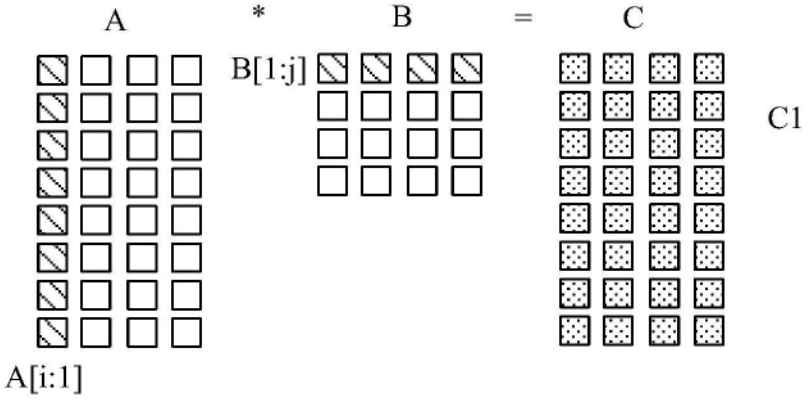


图2A

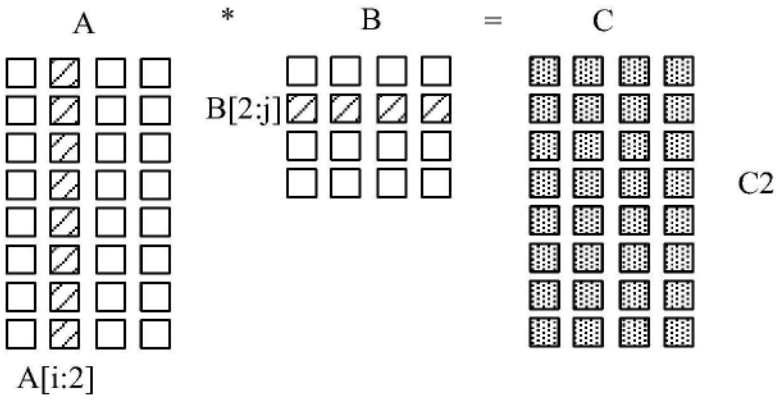


图2B

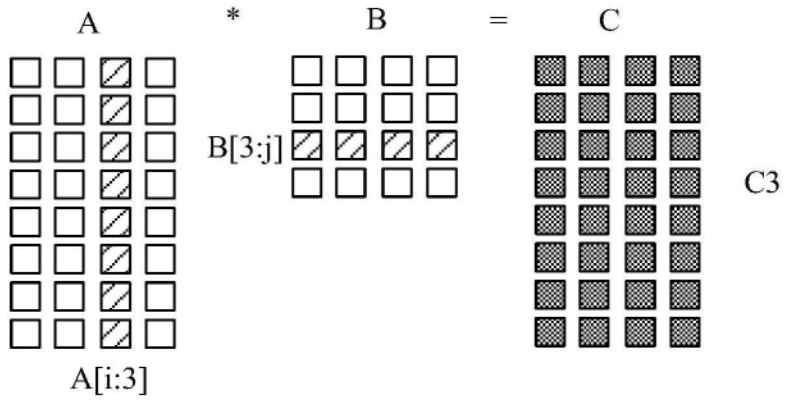


图2C

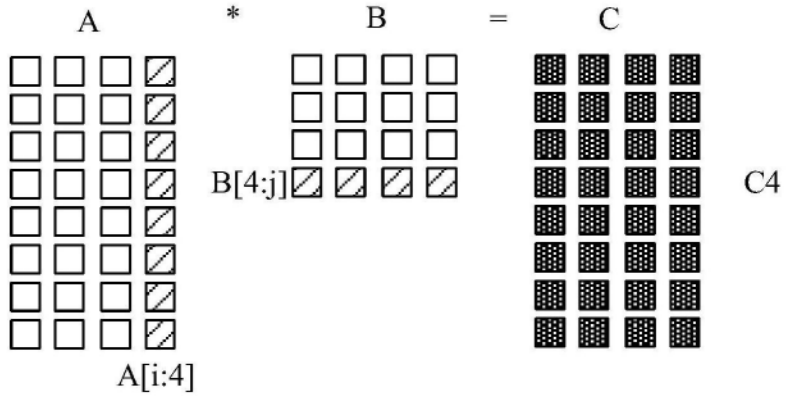


图2D

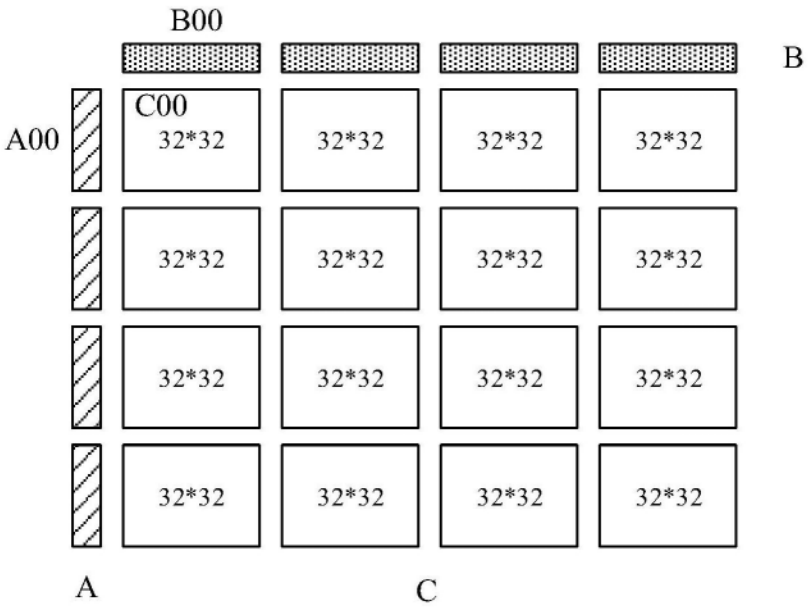


图3A

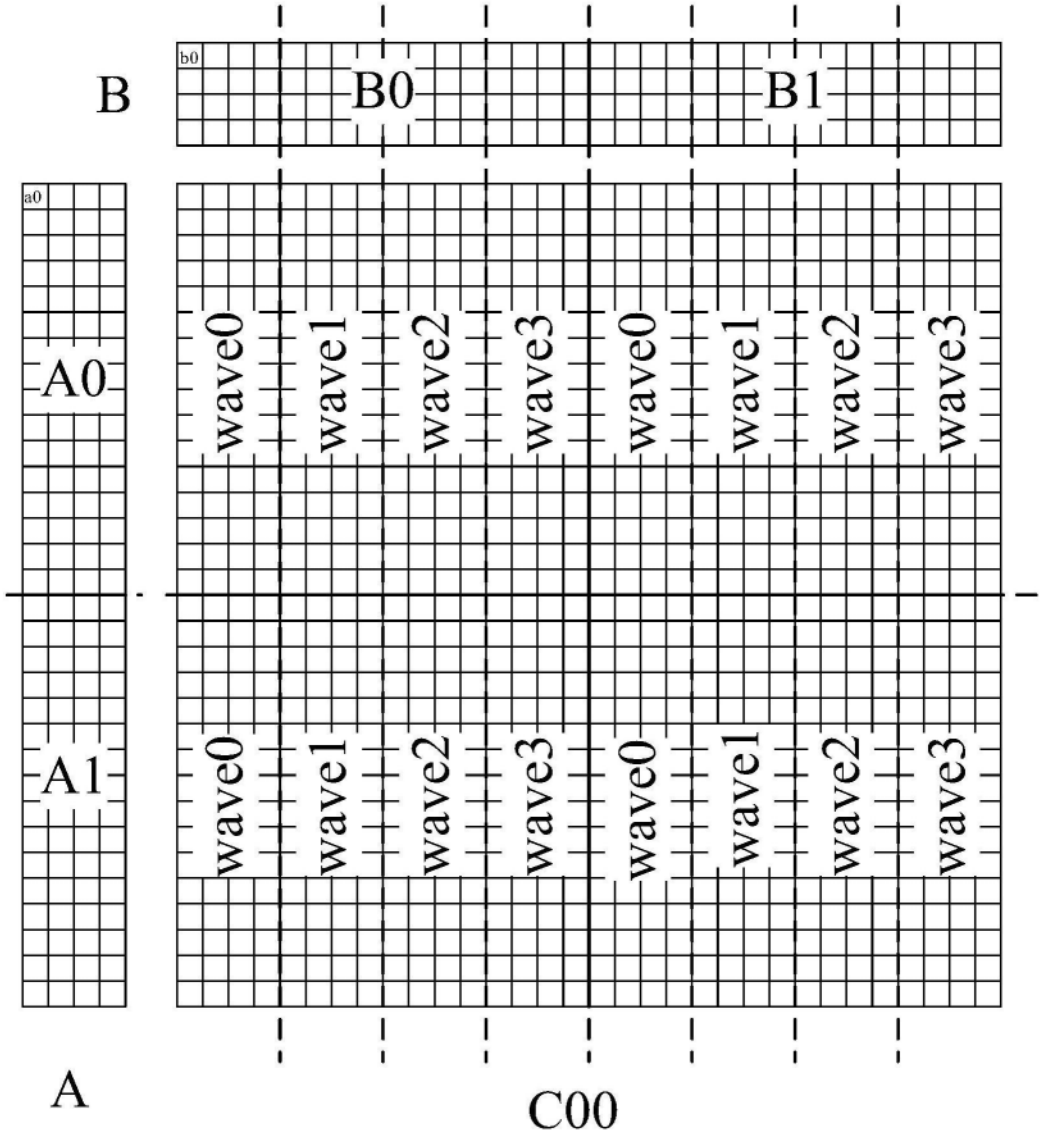


图3B

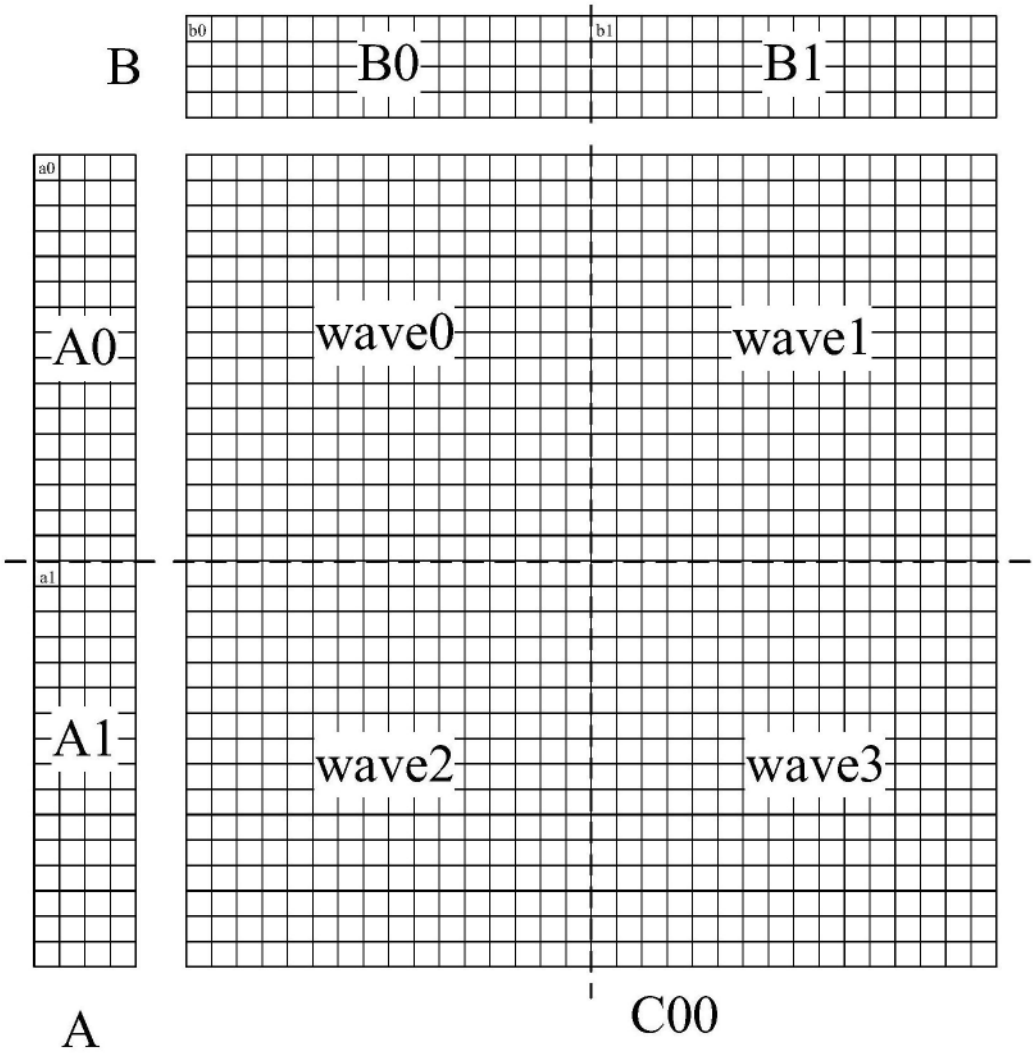


图3C

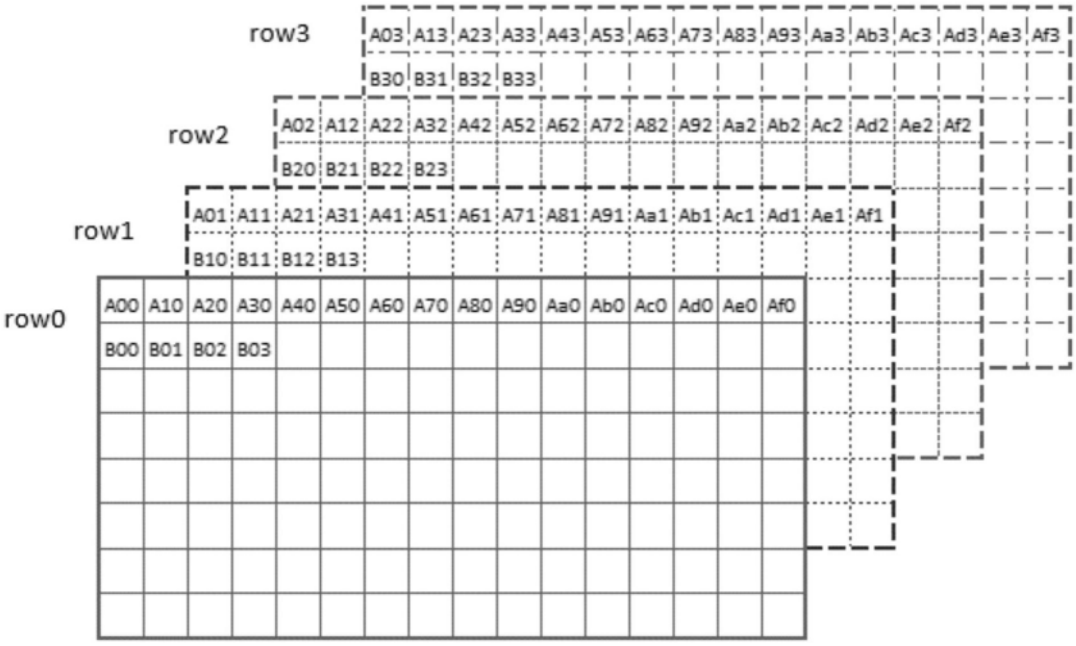


图3D

ROW0

v[0:1]	A00	A10	A20	A30	A40	A50	A60	A70	A80	A90	Aa0	Ab0	Ac0	Ad0	Ae0	Af0
v[2:3]	A01	A11	A12	A13	A14	A15	A16	A17	A18	A19	Aa1	Ab1	Ac1	Ad1	Ae1	Af1
v[4:5]	A02	A12	A22	A32	A42	A52	A62	A72	A82	A92	Aa2	Ab2	Ac2	Ad2	Ae2	Af2
v[6:7]	A03	A13	A23	A33	A43	A53	A63	A73	A83	A93	Aa3	Aa3	Ac3	Ad3	Ae3	Af3
v[80:81]	B00	B00	B00	B00	B00	B00	B00	B00	B00	B00	B00	B00	B00	B00	B00	B00
v[82:83]	B10	B10	B10	B10	B10	B10	B10	B10	B10	B10	B10	B10	B10	B10	B10	B10
v[84:85]	B20	B20	B20	B20	B20	B20	B20	B20	B20	B20	B20	B20	B20	B20	B20	B20
v[86:87]	B30	B30	B30	B30	B30	B30	B30	B30	B30	B30	B30	B30	B30	B30	B30	B30

图3E

ROW1

v[0:1]	A00	A10	A20	A30	A40	A50	A60	A70	A80	A90	Aa0	Ab0	Ac0	Ad0	Ae0	Af0
v[2:3]	A01	A11	A12	A13	A14	A15	A16	A17	A18	A19	Aa1	Ab1	Ac1	Ad1	Ae1	Af1
v[4:5]	A02	A12	A22	A32	A42	A52	A62	A72	A82	A92	Aa2	Ab2	Ac2	Ad2	Ae2	Af2
v[6:7]	A03	A13	A23	A33	A43	A53	A63	A73	A83	A93	Aa3	Aa3	Ac3	Ad3	Ae3	Af3
v[80:81]	B01	B01	B01	B01	B01	B01	B01	B01	B01	B01	B01	B01	B01	B01	B01	B01
v[82:83]	B11	B11	B11	B11	B11	B11	B11	B11	B11	B11	B11	B11	B11	B11	B11	B11
v[84:85]	B21	B21	B21	B21	B21	B21	B21	B21	B21	B21	B21	B21	B21	B21	B21	B21
v[86:87]	B31	B31	B31	B31	B31	B31	B31	B31	B31	B31	B31	B31	B31	B31	B31	B31

图3F

ROW2

v[0:1]	A00	A10	A20	A30	A40	A50	A60	A70	A80	A90	Aa0	Ab0	Ac0	Ad0	Ae0	Af0
v[2:3]	A01	A11	A12	A13	A14	A15	A16	A17	A18	A19	Aa1	Ab1	Ac1	Ad1	Ae1	Af1
v[4:5]	A02	A12	A22	A32	A42	A52	A62	A72	A82	A92	Aa2	Ab2	Ac2	Ad2	Ae2	Af2
v[6:7]	A03	A13	A23	A33	A43	A53	A63	A73	A83	A93	Aa3	Aa3	Ac3	Ad3	Ae3	Af3
v[80:81]	B02	B02	B02	B02	B02	B02	B02	B02	B02	B02	B02	B02	B02	B02	B02	B02
v[82:83]	B12	B12	B12	B12	B12	B12	B12	B12	B12	B12	B12	B12	B12	B12	B12	B12
v[84:85]	B22	B22	B22	B22	B22	B22	B22	B22	B22	B22	B22	B22	B22	B22	B22	B22
v[86:87]	B32	B32	B32	B32	B32	B32	B32	B32	B32	B32	B32	B32	B32	B32	B32	B32

图3G

ROW3

v[0:1]	A00	A10	A20	A30	A40	A50	A60	A70	A80	A90	Aa0	Ab0	Ac0	Ad0	Ae0	Af0
v[2:3]	A01	A11	A12	A13	A14	A15	A16	A17	A18	A19	Aa1	Ab1	Ac1	Ad1	Ae1	Af1
v[4:5]	A02	A12	A22	A32	A42	A52	A62	A72	A82	A92	Aa2	Ab2	Ac2	Ad2	Ae2	Af2
v[6:7]	A03	A13	A23	A33	A43	A53	A63	A73	A83	A93	Aa3	Aa3	Ac3	Ad3	Ae3	Af3
v[80:81]	B03	B03	B03	B03	B03	B03	B03	B03	B03	B03	B03	B03	B03	B03	B03	B03
v[82:83]	B13	B13	B13	B13	B13	B13	B13	B13	B13	B13	B13	B13	B13	B13	B13	B13
v[84:85]	B23	B23	B23	B23	B23	B23	B23	B23	B23	B23	B23	B23	B23	B23	B23	B23
v[86:87]	B33	B33	B33	B33	B33	B33	B33	B33	B33	B33	B33	B33	B33	B33	B33	B33

图3H

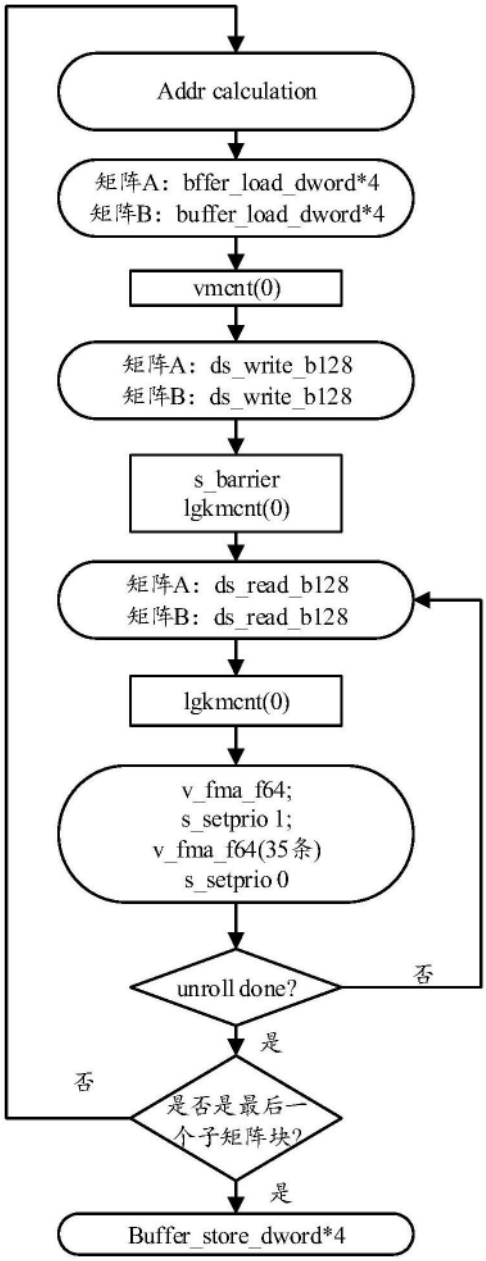

Addr calculation
矩阵A：bffer_load_dword*4
矩阵B：buffer_load_dword*4
vmcnt(0)
矩阵A：ds_write_b128
矩阵B：ds_write_b128
s_barrier
lgkmcnt(0)
矩阵A：ds_read_b128
矩阵B：ds_read_b128
lgkmcnt(0)
v_fma_f64;
s_setprio 1;
v_fma_f64(35条)
s_setprio 0
unroll done?
否
是
否
是否是最后一个子矩阵块?
是
Buffer_store_dword*4

图3I

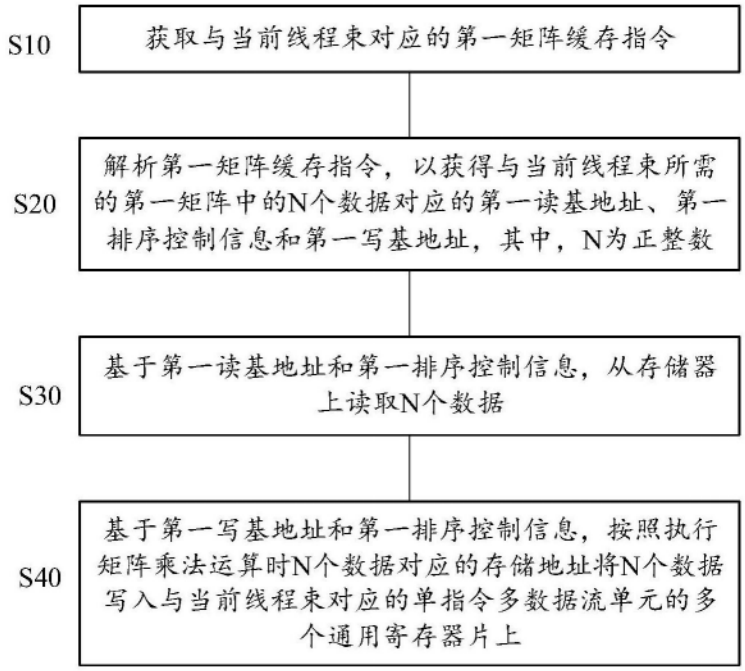

S10
获取与当前线程束对应的第一矩阵缓存指令
S20
解析第一矩阵缓存指令，以获得与当前线程束所需的第一矩阵中的N个数据对应的第一读基地址、第一排序控制信息和第一写基地址，其中，N为正整数
S30
基于第一读基地址和第一排序控制信息，从存储器上读取N个数据
S40
基于第一写基地址和第一排序控制信息，按照执行矩阵乘法运算时N个数据对应的存储地址将N个数据写入与当前线程束对应的单指令多数据流单元的多个通用寄存器片上

图4

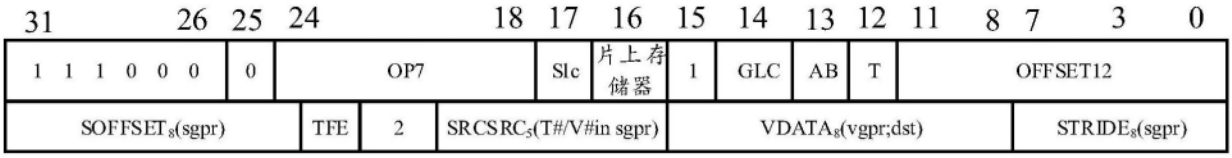

31 26 25 24 18 17 16 15 14 13 12 11 8 7 3 0
1 1 1 0 0 0
0
OP7
Slc
片上存储器
1
GLC
AB
T
OFFSET12
SOFFSET8(sgpr)
TFE
2
SRCSRC5(T#/V#in sgpr)
VDATA8(vgpr;dst)
STRIDE8(sgpr)

图5

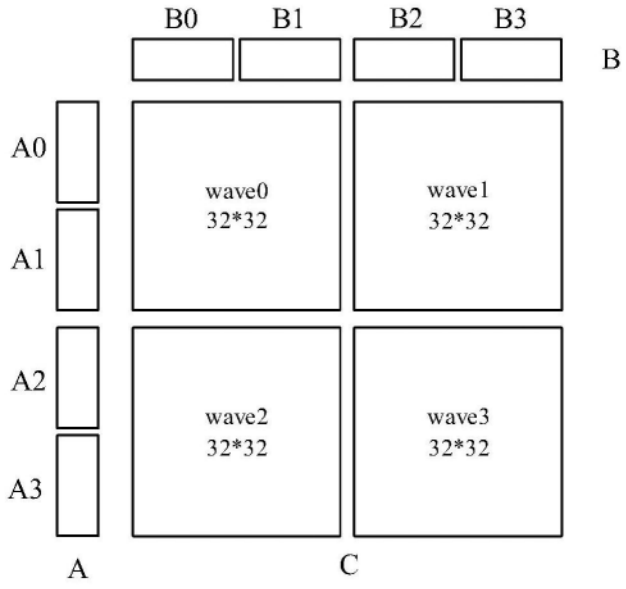

B0
B1
B2
B3
B
A0
A1
A2
A3
wave0
32*32
wave1
32*32
wave2
32*32
wave3
32*32
A
C

图6A

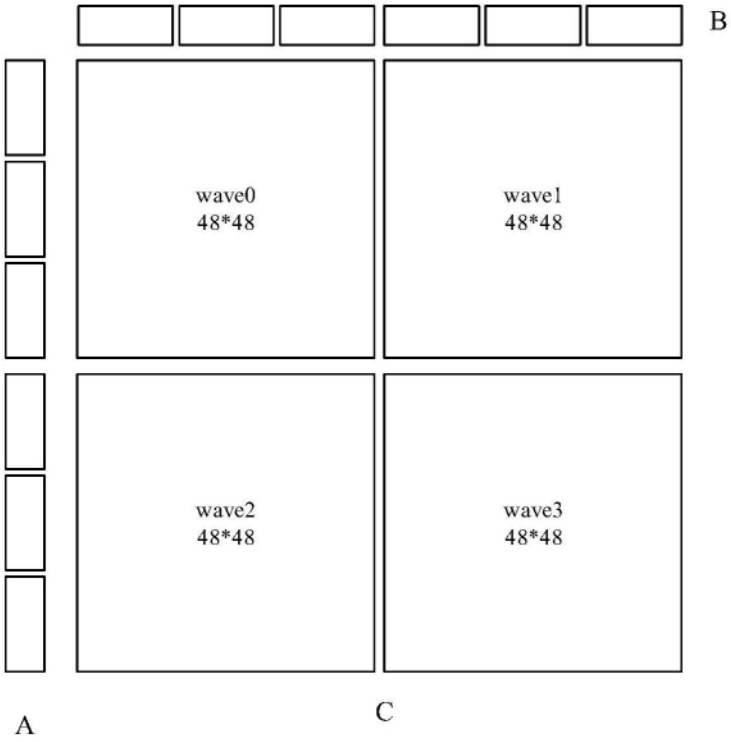


图6B

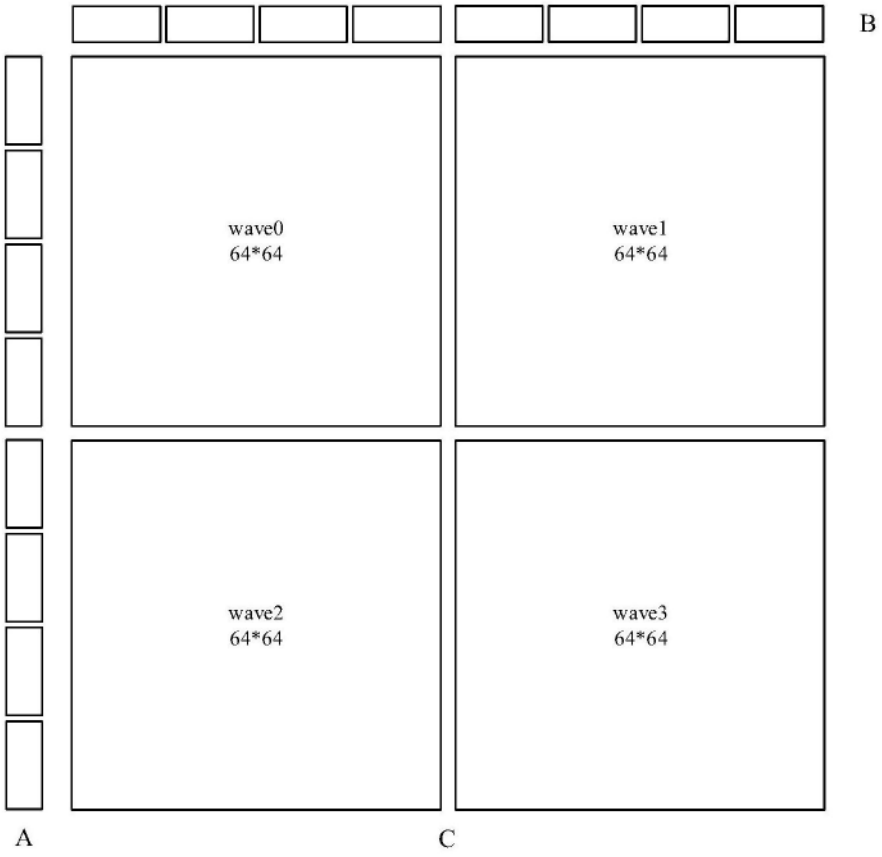


图6C

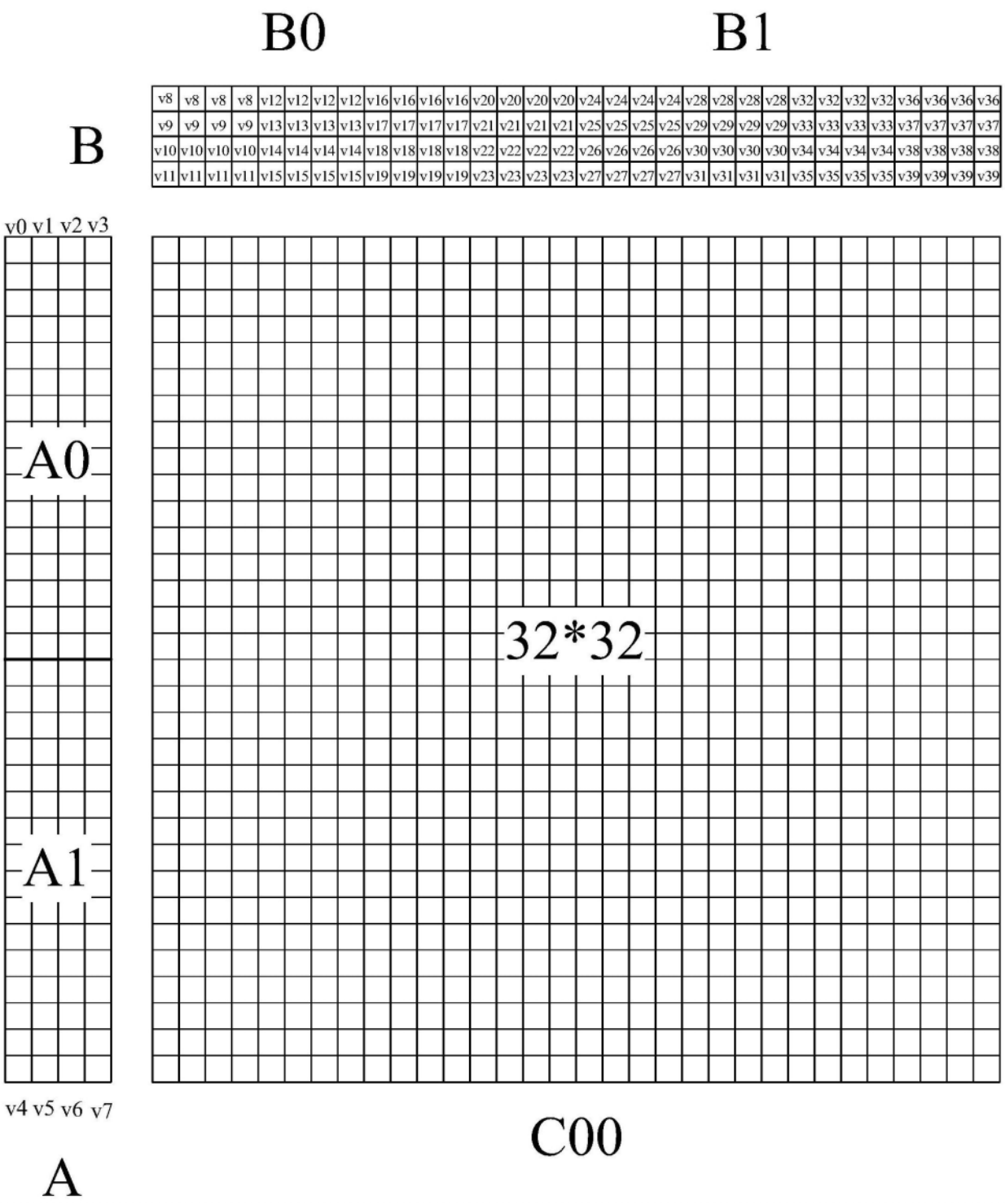

B0
B1
B
v8 v8 v8 v8 v12 v12 v12 v12 v16 v16 v16 v16 v20 v20 v20 v20 v24 v24 v24 v24 v28 v28 v28 v28 v32 v32 v32 v32 v36 v36 v36 v36
v9 v9 v9 v9 v13 v13 v13 v13 v17 v17 v17 v17 v21 v21 v21 v21 v25 v25 v25 v25 v29 v29 v29 v29 v33 v33 v33 v33 v37 v37 v37 v37
v10 v10 v10 v10 v14 v14 v14 v14 v18 v18 v18 v18 v22 v22 v22 v22 v26 v26 v26 v26 v30 v30 v30 v30 v34 v34 v34 v34 v38 v38 v38 v38
v11 v11 v11 v11 v15 v15 v15 v15 v19 v19 v19 v19 v23 v23 v23 v23 v27 v27 v27 v27 v31 v31 v31 v31 v35 v35 v35 v35 v39 v39 v39 v39
v0 v1 v2 v3
A0
32*32
A1
v4 v5 v6 v7
A
C00

图7A

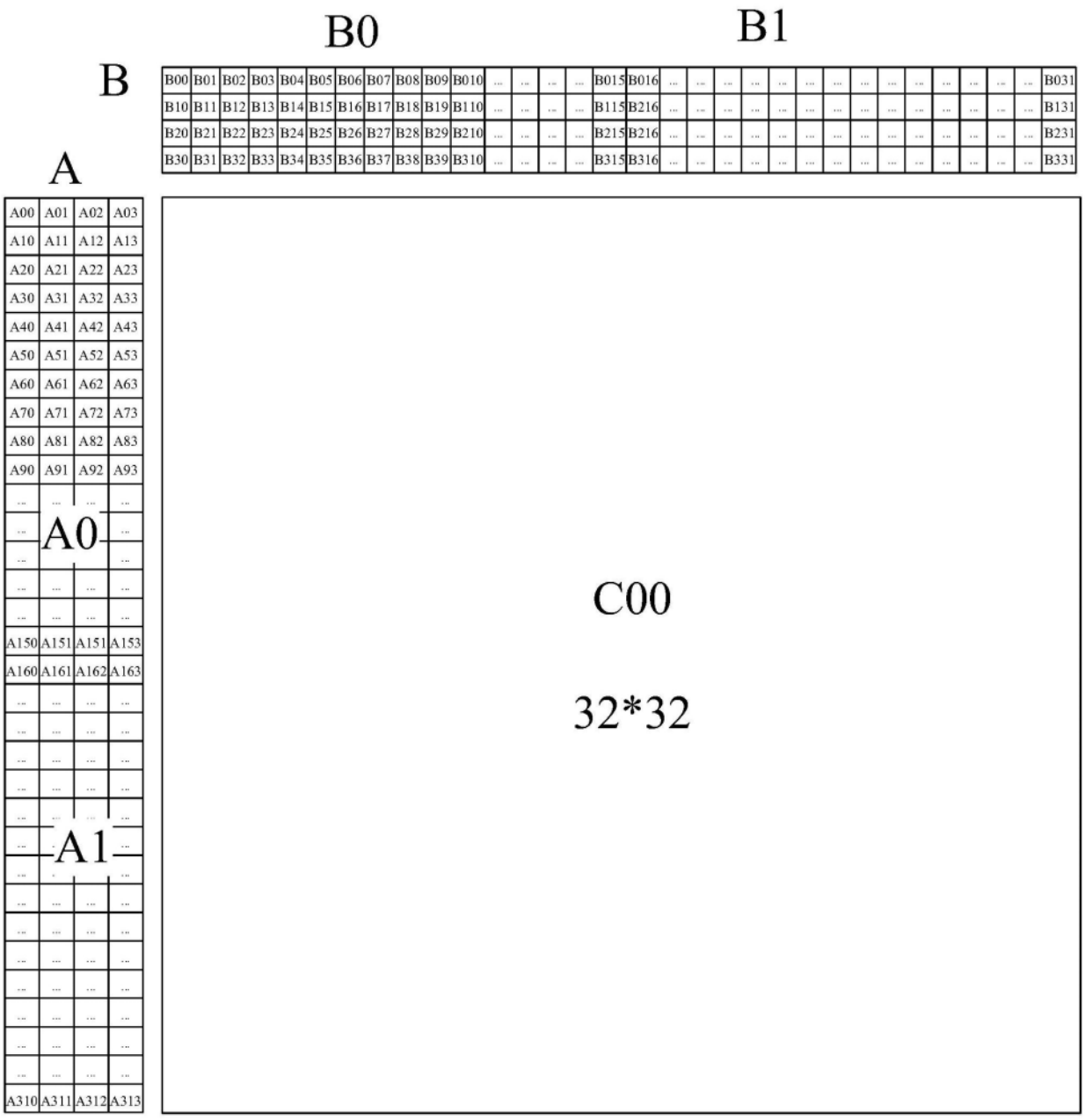


图7B

v0	A0,0	A1,0	A2,0	A3,0	A4,0	A5,0	A6,0	A7,0	A8,0	A9,0	A10,0	A11,0	A12,0	A13,0	A14,0	A15,0
v1																
v2																
v3																
v4	A16,0	A17,0	A18,0	A19,0	A20,0	A21,0	A22,0	A23,0	A24,0	A25,0	A26,0	A27,0	A28,0	A29,0	A30,0	A31,0
v5																
v6																
v7																
v8	B0,0	B0,0	B0,0	B0,0	B0,0	B0,0	B0,0	B0,0	B0,0	B0,0	B0,0	B0,0	B0,0	B0,0	B0,0	B0,0
v9	B1,0	B1,0	B1,0	B1,0	B1,0	B1,0	B1,0	B1,0	B1,0	B1,0	B1,0	B1,0	B1,0	B1,0	B1,0	B1,0
v10	B2,0	B2,0	B2,0	B2,0	B2,0	B2,0	B2,0	B2,0	B2,0	B2,0	B2,0	B2,0	B2,0	B2,0	B2,0	B2,0
v11	B3,0	B3,0	B3,0	B3,0	B3,0	B3,0	B3,0	B3,0	B3,0	B3,0	B3,0	B3,0	B3,0	B3,0	B3,0	B3,0
v12	B0,4	B0,4	B0,4	B0,4	B0,4	B0,4	B0,4	B0,4	B0,4	B0,4	B0,4	B0,4	B0,4	B0,4	B0,4	B0,4
v13	B1,4	B1,4	B1,4	B1,4	B1,4	B1,4	B1,4	B1,4	B1,4	B1,4	B1,4	B1,4	B1,4	B1,4	B1,4	B1,4
v14	B2,4	B2,4	B2,4	B2,4	B2,4	B2,4	B2,4	B2,4	B2,4	B2,4	B2,4	B2,4	B2,4	B2,4	B2,4	B2,4
v15	B3,4	B3,4	B3,4	B3,4	B3,4	B3,4	B3,4	B3,4	B3,4	B3,4	B3,4	B3,4	B3,4	B3,4	B3,4	B3,4
v16	B0,8	B0,8	B0,8	B0,8	B0,8	B0,8	B0,8	B0,8	B0,8	B0,8	B0,8	B0,8	B0,8	B0,8	B0,8	B0,8
v17	B1,8	B1,8	B1,8	B1,8	B1,8	B1,8	B1,8	B1,8	B1,8	B1,8	B1,8	B1,8	B1,8	B1,8	B1,8	B1,8
v18	B2,8	B2,8	B2,8	B2,8	B2,8	B2,8	B2,8	B2,8	B2,8	B2,8	B2,8	B2,8	B2,8	B2,8	B2,8	B2,8
v19	B3,8	B3,8	B3,8	B3,8	B3,8	B3,8	B3,8	B3,8	B3,8	B3,8	B3,8	B3,8	B3,8	B3,8	B3,8	B3,8
v20	B0,12	B0,12	B0,12	B0,12	B0,12	B0,12	B0,12	B0,12	B0,12	B0,12	B0,12	B0,12	B0,12	B0,12	B0,12	B0,12
v21	B1,12	B1,12	B1,12	B1,12	B1,12	B1,12	B1,12	B1,12	B1,12	B1,12	B1,12	B1,12	B1,12	B1,12	B1,12	B1,12
v22	B2,12	B2,12	B2,12	B2,12	B2,12	B2,12	B2,12	B2,12	B2,12	B2,12	B2,12	B2,12	B2,12	B2,12	B2,12	B2,12
v23	B3,12	B3,12	B3,12	B3,12	B3,12	B3,12	B3,12	B3,12	B3,12	B3,12	B3,12	B3,12	B3,12	B3,12	B3,12	B3,12
v24	B0,16	B0,16	B0,16	B0,16	B0,16	B0,16	B0,16	B0,16	B0,16	B0,16	B0,16	B0,16	B0,16	B0,16	B0,16	B0,16
v25	B1,16	B1,16	B1,16	B1,16	B1,16	B1,16	B1,16	B1,16	B1,16	B1,16	B1,16	B1,16	B1,16	B1,16	B1,16	B1,16
v26	B2,16	B2,16	B2,16	B2,16	B2,16	B2,16	B2,16	B2,16	B2,16	B2,16	B2,16	B2,16	B2,16	B2,16	B2,16	B2,16
v27	B3,16	B3,16	B3,16	B3,16	B3,16	B3,16	B3,16	B3,16	B3,16	B3,16	B3,16	B3,16	B3,16	B3,16	B3,16	B3,16
v28	B0,20	B0,20	B0,20	B0,20	B0,20	B0,20	B0,20	B0,20	B0,20	B0,20	B0,20	B0,20	B0,20	B0,20	B0,20	B0,20
v29	B1,20	B1,20	B1,20	B1,20	B1,20	B1,20	B1,20	B1,20	B1,20	B1,20	B1,20	B1,20	B1,20	B1,20	B1,20	B1,20
v30	B2,20	B2,20	B2,20	B2,20	B2,20	B2,20	B2,20	B2,20	B2,20	B2,20	B2,20	B2,20	B2,20	B2,20	B2,20	B2,20
v31	B3,20	B3,20	B3,20	B3,20	B3,20	B3,20	B3,20	B3,20	B3,20	B3,20	B3,20	B3,20	B3,20	B3,20	B3,20	B3,20
v32	B0,24	B0,24	B0,24	B0,24	B0,24	B0,24	B0,24	B0,24	B0,24	B0,24	B0,24	B0,24	B0,24	B0,24	B0,24	B0,24
v33	B1,24	B1,24	B1,24	B1,24	B1,24	B1,24	B1,24	B1,24	B1,24	B1,24	B1,24	B1,24	B1,24	B1,24	B1,24	B1,24
v34	B2,24	B2,24	B2,24	B2,24	B2,24	B2,24	B2,24	B2,24	B2,24	B2,24	B2,24	B2,24	B2,24	B2,24	B2,24	B2,24
v35	B3,24	B3,24	B3,24	B3,24	B3,24	B3,24	B3,24	B3,24	B3,24	B3,24	B3,24	B3,24	B3,24	B3,24	B3,24	B3,24
v36	B0,28	B0,28	B0,28	B0,28	B0,28	B0,28	B0,28	B0,28	B0,28	B0,28	B0,28	B0,28	B0,28	B0,28	B0,28	B0,28
v37	B1,28	B1,28	B1,28	B1,28	B1,28	B1,28	B1,28	B1,28	B1,28	B1,28	B1,28	B1,28	B1,28	B1,28	B1,28	B1,28
v38	B2,28	B2,28	B2,28	B2,28	B2,28	B2,28	B2,28	B2,28	B2,28	B2,28	B2,28	B2,28	B2,28	B2,28	B2,28	B2,28
v39	B3,28	B3,28	B3,28	B3,28	B3,28	B3,28	B3,28	B3,28	B3,28	B3,28	B3,28	B3,28	B3,28	B3,28	B3,28	B3,28

图8A

v0																
v1	A0,1	A1,1	A2,1	A3,1	A4,1	A5,1	A6,1	A7,1	A8,1	A9,1	A10,1	A11,1	A12,1	A13,1	A14,1	A15,1
v2																
v3																
v4																
v5	A16,1	A17,1	A18,1	A19,1	A20,1	A21,1	A22,1	A23,1	A24,1	A25,1	A26,1	A27,1	A28,1	A29,1	A30,1	A31,1
v6																
v7																
v8	B0,1	B0,1	B0,1	B0,1	B0,1	B0,1	B0,1	B0,1	B0,1	B0,1	B0,1	B0,1	B0,1	B0,1	B0,1	B0,1
v9	B1,1	B1,1	B1,1	B1,1	B1,1	B1,1	B1,1	B1,1	B1,1	B1,1	B1,1	B1,1	B1,1	B1,1	B1,1	B1,1
v10	B2,1	B2,1	B2,1	B2,1	B2,1	B2,1	B2,1	B2,1	B2,1	B2,1	B2,1	B2,1	B2,1	B2,1	B2,1	B2,1
v11	B3,1	B3,1	B3,1	B3,1	B3,1	B3,1	B3,1	B3,1	B3,1	B3,1	B3,1	B3,1	B3,1	B3,1	B3,1	B3,1
v12	B0,5	B0,5	B0,5	B0,5	B0,5	B0,5	B0,5	B0,5	B0,5	B0,5	B0,5	B0,5	B0,5	B0,5	B0,5	B0,5
v13	B1,5	B1,5	B1,5	B1,5	B1,5	B1,5	B1,5	B1,5	B1,5	B1,5	B1,5	B1,5	B1,5	B1,5	B1,5	B1,5
v14	B2,5	B2,5	B2,5	B2,5	B2,5	B2,5	B2,5	B2,5	B2,5	B2,5	B2,5	B2,5	B2,5	B2,5	B2,5	B2,5
v15	B3,5	B3,5	B3,5	B3,5	B3,5	B3,5	B3,5	B3,5	B3,5	B3,5	B3,5	B3,5	B3,5	B3,5	B3,5	B3,5
v16	B0,9	B0,9	B0,9	B0,9	B0,9	B0,9	B0,9	B0,9	B0,9	B0,9	B0,9	B0,9	B0,9	B0,9	B0,9	B0,9
v17	B1,9	B1,9	B1,9	B1,9	B1,9	B1,9	B1,9	B1,9	B1,9	B1,9	B1,9	B1,9	B1,9	B1,9	B1,9	B1,9
v18	B2,9	B2,9	B2,9	B2,9	B2,9	B2,9	B2,9	B2,9	B2,9	B2,9	B2,9	B2,9	B2,9	B2,9	B2,9	B2,9
v19	B3,9	B3,9	B3,9	B3,9	B3,9	B3,9	B3,9	B3,9	B3,9	B3,9	B3,9	B3,9	B3,9	B3,9	B3,9	B3,9
v20	B0,13	B0,13	B0,13	B0,13	B0,13	B0,13	B0,13	B0,13	B0,13	B0,13	B0,13	B0,13	B0,13	B0,13	B0,13	B0,13
v21	B1,13	B1,13	B1,13	B1,13	B1,13	B1,13	B1,13	B1,13	B1,13	B1,13	B1,13	B1,13	B1,13	B1,13	B1,13	B1,13
v22	B2,13	B2,13	B2,13	B2,13	B2,13	B2,13	B2,13	B2,13	B2,13	B2,13	B2,13	B2,13	B2,13	B2,13	B2,13	B2,13
v23	B3,13	B3,13	B3,13	B3,13	B3,13	B3,13	B3,13	B3,13	B3,13	B3,13	B3,13	B3,13	B3,13	B3,13	B3,13	B3,13
v24	B0,17	B0,17	B0,17	B0,17	B0,17	B0,17	B0,17	B0,17	B0,17	B0,17	B0,17	B0,17	B0,17	B0,17	B0,17	B0,17
v25	B1,17	B1,17	B1,17	B1,17	B1,17	B1,17	B1,17	B1,17	B1,17	B1,17	B1,17	B1,17	B1,17	B1,17	B1,17	B1,17
v26	B2,17	B2,17	B2,17	B2,17	B2,17	B2,17	B2,17	B2,17	B2,17	B2,17	B2,17	B2,17	B2,17	B2,17	B2,17	B2,17
v27	B3,17	B3,17	B3,17	B3,17	B3,17	B3,17	B3,17	B3,17	B3,17	B3,17	B3,17	B3,17	B3,17	B3,17	B3,17	B3,17
v28	B0,21	B0,21	B0,21	B0,21	B0,21	B0,21	B0,21	B0,21	B0,21	B0,21	B0,21	B0,21	B0,21	B0,21	B0,21	B0,21
v29	B1,21	B1,21	B1,21	B1,21	B1,21	B1,21	B1,21	B1,21	B1,21	B1,21	B1,21	B1,21	B1,21	B1,21	B1,21	B1,21
v30	B2,21	B2,21	B2,21	B2,21	B2,21	B2,21	B2,21	B2,21	B2,21	B2,21	B2,21	B2,21	B2,21	B2,21	B2,21	B2,21
v31	B3,21	B3,21	B3,21	B3,21	B3,21	B3,21	B3,21	B3,21	B3,21	B3,21	B3,21	B3,21	B3,21	B3,21	B3,21	B3,21
v32	B0,25	B0,25	B0,25	B0,25	B0,25	B0,25	B0,25	B0,25	B0,25	B0,25	B0,25	B0,25	B0,25	B0,25	B0,25	B0,25
v33	B1,25	B1,25	B1,25	B1,25	B1,25	B1,25	B1,25	B1,25	B1,25	B1,25	B1,25	B1,25	B1,25	B1,25	B1,25	B1,25
v34	B2,25	B2,25	B2,25	B2,25	B2,25	B2,25	B2,25	B2,25	B2,25	B2,25	B2,25	B2,25	B2,25	B2,25	B2,25	B2,25
v35	B3,25	B3,25	B3,25	B3,25	B3,25	B3,25	B3,25	B3,25	B3,25	B3,25	B3,25	B3,25	B3,25	B3,25	B3,25	B3,25
v36	B0,29	B0,29	B0,29	B0,29	B0,29	B0,29	B0,29	B0,29	B0,29	B0,29	B0,29	B0,29	B0,29	B0,29	B0,29	B0,29
v37	B1,29	B1,29	B1,29	B1,29	B1,29	B1,29	B1,29	B1,29	B1,29	B1,29	B1,29	B1,29	B1,29	B1,29	B1,29	B1,29
v38	B2,29	B2,29	B2,29	B2,29	B2,29	B2,29	B2,29	B2,29	B2,29	B2,29	B2,29	B2,29	B2,29	B2,29	B2,29	B2,29
v39	B3,29	B3,29	B3,29	B3,29	B3,29	B3,29	B3,29	B3,29	B3,29	B3,29	B3,29	B3,29	B3,29	B3,29	B3,29	B3,29

图8B

v0															
v1															
v2	A0,2	A1,2	A2,2	A3,2	A4,2	A5,2	A6,2	A7,2	A8,2	A9,2	A10,2	A11,2	A12,2	A13,2	A14,2	A15,2
v3																
v4																
v5																
v6	A16,2	A17,2	A18,2	A19,2	A20,2	A21,2	A22,2	A23,2	A24,2	A25,2	A26,2	A27,2	A28,2	A29,2	A30,2	A31,2
v7																
v8	B0,2	B02	B02	B02	B02	B02	B02	B02	B02	B02	B02	B02	B02	B02	B02	B02
v9	B1,2	B1,2	B1,2	B1,2	B1,2	B1,2	B1,2	B1,2	B1,2	B1,2	B1,2	B1,2	B1,2	B1,2	B1,2	B1,2
v10	B2,2	B2,2	B2,2	B2,2	B2,2	B2,2	B2,2	B2,2	B2,2	B2,2	B2,2	B2,2	B2,2	B2,2	B2,2	B2,2
v11	B3,2	B3,2	B3,2	B3,2	B3,2	B3,2	B3,2	B3,2	B3,2	B3,2	B3,2	B3,2	B3,2	B3,2	B3,2	B3,2
v12	B0,6	B0,6	B0,6	B0,6	B0,6	B0,6	B0,6	B0,6	B0,6	B0,6	B0,6	B0,6	B0,6	B0,6	B0,6	B0,6
v13	B1,6	B1,6	B1,6	B1,6	B1,6	B1,6	B1,6	B1,6	B1,6	B1,6	B1,6	B1,6	B1,6	B1,6	B1,6	B1,6
v14	B2,6	B2,6	B2,6	B2,6	B2,6	B2,6	B2,6	B2,6	B2,6	B2,6	B2,6	B2,6	B2,6	B2,6	B2,6	B2,6
v15	B3,6	B3,6	B3,6	B3,6	B3,6	B3,6	B3,6	B3,6	B3,6	B3,6	B3,6	B3,6	B3,6	B3,6	B3,6	B3,6
v16	B0,10	B0,10	B0,10	B0,10	B0,10	B0,10	B0,10	B0,10	B0,10	B0,10	B0,10	B0,10	B0,10	B0,10	B0,10	B0,10
v17	B1,10	B1,10	B1,10	B1,10	B1,10	B1,10	B1,10	B1,10	B1,10	B1,10	B1,10	B1,10	B1,10	B1,10	B1,10	B1,10
v18	B2,10	B2,10	B2,10	B2,10	B2,10	B2,10	B2,10	B2,10	B2,10	B2,10	B2,10	B2,10	B2,10	B2,10	B2,10	B2,10
v19	B3,10	B3,10	B3,10	B3,10	B3,10	B3,10	B3,10	B3,10	B3,10	B3,10	B3,10	B3,10	B3,10	B3,10	B3,10	B3,10
v20	B0,14	B0,14	B0,14	B0,14	B0,14	B0,14	B0,14	B0,14	B0,14	B0,14	B0,14	B0,14	B0,14	B0,14	B0,14	B0,14
v21	B1,14	B1,14	B1,14	B1,14	B1,14	B1,14	B1,14	B1,14	B1,14	B1,14	B1,14	B1,14	B1,14	B1,14	B1,14	B1,14
v22	B2,14	B2,14	B2,14	B2,14	B2,14	B2,14	B2,14	B2,14	B2,14	B2,14	B2,14	B2,14	B2,14	B2,14	B2,14	B2,14
v23	B3,14	B3,14	B3,14	B3,14	B3,14	B3,14	B3,14	B3,14	B3,14	B3,14	B3,14	B3,14	B3,14	B3,14	B3,14	B3,14
v24	B0,18	B0,18	B0,18	B0,18	B0,18	B0,18	B0,18	B0,18	B0,18	B0,18	B0,18	B0,18	B0,18	B0,18	B0,18	B0,18
v25	B1,18	B1,18	B1,18	B1,18	B1,18	B1,18	B1,18	B1,18	B1,18	B1,18	B1,18	B1,18	B1,18	B1,18	B1,18	B1,18
v26	B2,18	B2,18	B2,18	B2,18	B2,18	B2,18	B2,18	B2,18	B2,18	B2,18	B2,18	B2,18	B2,18	B2,18	B2,18	B2,18
v27	B3,18	B3,18	B3,18	B3,18	B3,18	B3,18	B3,18	B3,18	B3,18	B3,18	B3,18	B3,18	B3,18	B3,18	B3,18	B3,18
v28	B0,22	B0,22	B0,22	B0,22	B0,22	B0,22	B0,22	B0,22	B0,22	B0,22	B0,22	B0,22	B0,22	B0,22	B0,22	B0,22
v29	B1,22	B1,22	B1,22	B1,22	B1,22	B1,22	B1,22	B1,22	B1,22	B1,22	B1,22	B1,22	B1,22	B1,22	B1,22	B1,22
v30	B2,22	B2,22	B2,22	B2,22	B2,22	B2,22	B2,22	B2,22	B2,22	B2,22	B2,22	B2,22	B2,22	B2,22	B2,22	B2,22
v31	B3,22	B3,22	B3,22	B3,22	B3,22	B3,22	B3,22	B3,22	B3,22	B3,22	B3,22	B3,22	B3,22	B3,22	B3,22	B3,22
v32	B0,26	B0,26	B0,26	B0,26	B0,26	B0,26	B0,26	B0,26	B0,26	B0,26	B0,26	B0,26	B0,26	B0,26	B0,26	B0,26
v33	B1,26	B1,26	B1,26	B1,26	B1,26	B1,26	B1,26	B1,26	B1,26	B1,26	B1,26	B1,26	B1,26	B1,26	B1,26	B1,26
v34	B2,26	B2,26	B2,26	B2,26	B2,26	B2,26	B2,26	B2,26	B2,26	B2,26	B2,26	B2,26	B2,26	B2,26	B2,26	B2,26
v35	B3,26	B3,26	B3,26	B3,26	B3,26	B3,26	B3,26	B3,26	B3,26	B3,26	B3,26	B3,26	B3,26	B3,26	B3,26	B3,26
v36	B0,30	B0,30	B0,30	B0,30	B0,30	B0,30	B0,30	B0,30	B0,30	B0,30	B0,30	B0,30	B0,30	B0,30	B0,30	B0,30
v37	B1,30	B1,30	B1,30	B1,30	B1,30	B1,30	B1,30	B1,30	B1,30	B1,30	B1,30	B1,30	B1,30	B1,30	B1,30	B1,30
v38	B2,30	B2,30	B2,30	B2,30	B2,30	B2,30	B2,30	B2,30	B2,30	B2,30	B2,30	B2,30	B2,30	B2,30	B2,30	B2,30
v39	B3,30	B3,30	B3,30	B3,30	B3,30	B3,30	B3,30	B3,30	B3,30	B3,30	B3,30	B3,30	B3,30	B3,30	B3,30	B3,30

图8C

v0																
v1																
v2																
v3	A0,3	A1,3	A2,3	A3,3	A4,3	A5,3	A6,3	A7,3	A8,3	A9,3	A10,3	A11,3	A12,3	A13,3	A14,3	A15,3
v4																
v5																
v6																
v7	A16,3	A17,3	A18,3	A19,3	A20,3	A21,3	A22,3	A23,3	A24,3	A25,3	A26,3	A27,3	A28,3	A29,3	A30,3	A31,3
v8	B0,3	B0,3	B0,3	B0,3	B0,3	B0,3	B0,3	B0,3	B0,3	B0,3	B0,3	B0,3	B0,3	B0,3	B0,3	B0,3
v9	B1,3	B1,3	B1,3	B1,3	B1,3	B1,3	B1,3	B1,3	B1,3	B1,3	B1,3	B1,3	B1,3	B1,3	B1,3	B1,3
v10	B2,3	B2,3	B2,3	B2,3	B2,3	B2,3	B2,3	B2,3	B2,3	B2,3	B2,3	B2,3	B2,3	B2,3	B2,3	B2,3
v11	B3,3	B3,3	B3,3	B3,3	B3,3	B3,3	B3,3	B3,3	B3,3	B3,3	B3,3	B3,3	B3,3	B3,3	B3,3	B3,3
v12	B0,7	B0,7	B0,7	B0,7	B0,7	B0,7	B0,7	B0,7	B0,7	B0,7	B0,7	B0,7	B0,7	B0,7	B0,7	B0,7
v13	B1,7	B1,7	B1,7	B1,7	B1,7	B1,7	B1,7	B1,7	B1,7	B1,7	B1,7	B1,7	B1,7	B1,7	B1,7	B1,7
v14	B2,7	B2,7	B2,7	B2,7	B2,7	B2,7	B2,7	B2,7	B2,7	B2,7	B2,7	B2,7	B2,7	B2,7	B2,7	B2,7
v15	B3,7	B3,7	B3,7	B3,7	B3,7	B3,7	B3,7	B3,7	B3,7	B3,7	B3,7	B3,7	B3,7	B3,7	B3,7	B3,7
v16	B0,11	B0,11	B0,11	B0,11	B0,11	B0,11	B0,11	B0,11	B0,11	B0,11	B0,11	B0,11	B0,11	B0,11	B0,11	B0,11
v17	B1,11	B1,11	B1,11	B1,11	B1,11	B1,11	B1,11	B1,11	B1,11	B1,11	B1,11	B1,11	B1,11	B1,11	B1,11	B1,11
v18	B2,11	B2,11	B2,11	B2,11	B2,11	B2,11	B2,11	B2,11	B2,11	B2,11	B2,11	B2,11	B2,11	B2,11	B2,11	B2,11
v19	B3,11	B3,11	B3,11	B3,11	B3,11	B3,11	B3,11	B3,11	B3,11	B3,11	B3,11	B3,11	B3,11	B3,11	B3,11	B3,11
v20	B0,15	B0,15	B0,15	B0,15	B0,15	B0,15	B0,15	B0,15	B0,15	B0,15	B0,15	B0,15	B0,15	B0,15	B0,15	B0,15
v21	B1,15	B1,15	B1,15	B1,15	B1,15	B1,15	B1,15	B1,15	B1,15	B1,15	B1,15	B1,15	B1,15	B1,15	B1,15	B1,15
v22	B2,15	B2,15	B2,15	B2,15	B2,15	B2,15	B2,15	B2,15	B2,15	B2,15	B2,15	B2,15	B2,15	B2,15	B2,15	B2,15
v23	B3,15	B3,15	B3,15	B3,15	B3,15	B3,15	B3,15	B3,15	B3,15	B3,15	B3,15	B3,15	B3,15	B3,15	B3,15	B3,15
v24	B0,19	B0,19	B0,19	B0,19	B0,19	B0,19	B0,19	B0,19	B0,19	B0,19	B0,19	B0,19	B0,19	B0,19	B0,19	B0,19
v25	B1,19	B1,19	B1,19	B1,19	B1,19	B1,19	B1,19	B1,19	B1,19	B1,19	B1,19	B1,19	B1,19	B1,19	B1,19	B1,19
v26	B2,19	B2,19	B2,19	B2,19	B2,19	B2,19	B2,19	B2,19	B2,19	B2,19	B2,19	B2,19	B2,19	B2,19	B2,19	B2,19
v27	B3,19	B3,19	B3,19	B3,19	B3,19	B3,19	B3,19	B3,19	B3,19	B3,19	B3,19	B3,19	B3,19	B3,19	B3,19	B3,19
v28	B0,23	B0,23	B0,23	B0,23	B0,23	B0,23	B0,23	B0,23	B0,23	B0,23	B0,23	B0,23	B0,23	B0,23	B0,23	B0,23
v29	B1,23	B1,23	B1,23	B1,23	B1,23	B1,23	B1,23	B1,23	B1,23	B1,23	B1,23	B1,23	B1,23	B1,23	B1,23	B1,23
v30	B2,23	B2,23	B2,23	B2,23	B2,23	B2,23	B2,23	B2,23	B2,23	B2,23	B2,23	B2,23	B2,23	B2,23	B2,23	B2,23
v31	B3,23	B3,23	B3,23	B3,23	B3,23	B3,23	B3,23	B3,23	B3,23	B3,23	B3,23	B3,23	B3,23	B3,23	B3,23	B3,23
v32	B0,27	B0,27	B0,27	B0,27	B0,27	B0,27	B0,27	B0,27	B0,27	B0,27	B0,27	B0,27	B0,27	B0,27	B0,27	B0,27
v33	B1,27	B1,27	B1,27	B1,27	B1,27	B1,27	B1,27	B1,27	B1,27	B1,27	B1,27	B1,27	B1,27	B1,27	B1,27	B1,27
v34	B2,27	B2,27	B2,27	B2,27	B2,27	B2,27	B2,27	B2,27	B2,27	B2,27	B2,27	B2,27	B2,27	B2,27	B2,27	B2,27
v35	B3,27	B3,27	B3,27	B3,27	B3,27	B3,27	B3,27	B3,27	B3,27	B3,27	B3,27	B3,27	B3,27	B3,27	B3,27	B3,27
v36	B0,31	B0,31	B0,31	B0,31	B0,31	B0,31	B0,31	B0,31	B0,31	B0,31	B0,31	B0,31	B0,31	B0,31	B0,31	B0,31
v37	B1,31	B1,31	B1,31	B1,31	B1,31	B1,31	B1,31	B1,31	B1,31	B1,31	B1,31	B1,31	B1,31	B1,31	B1,31	B1,31
v38	B2,31	B2,31	B2,31	B2,31	B2,31	B2,31	B2,31	B2,31	B2,31	B2,31	B2,31	B2,31	B2,31	B2,31	B2,31	B2,31
v39	B3,31	B3,31	B3,31	B3,31	B3,31	B3,31	B3,31	B3,31	B3,31	B3,31	B3,31	B3,31	B3,31	B3,31	B3,31	B3,31

图8D

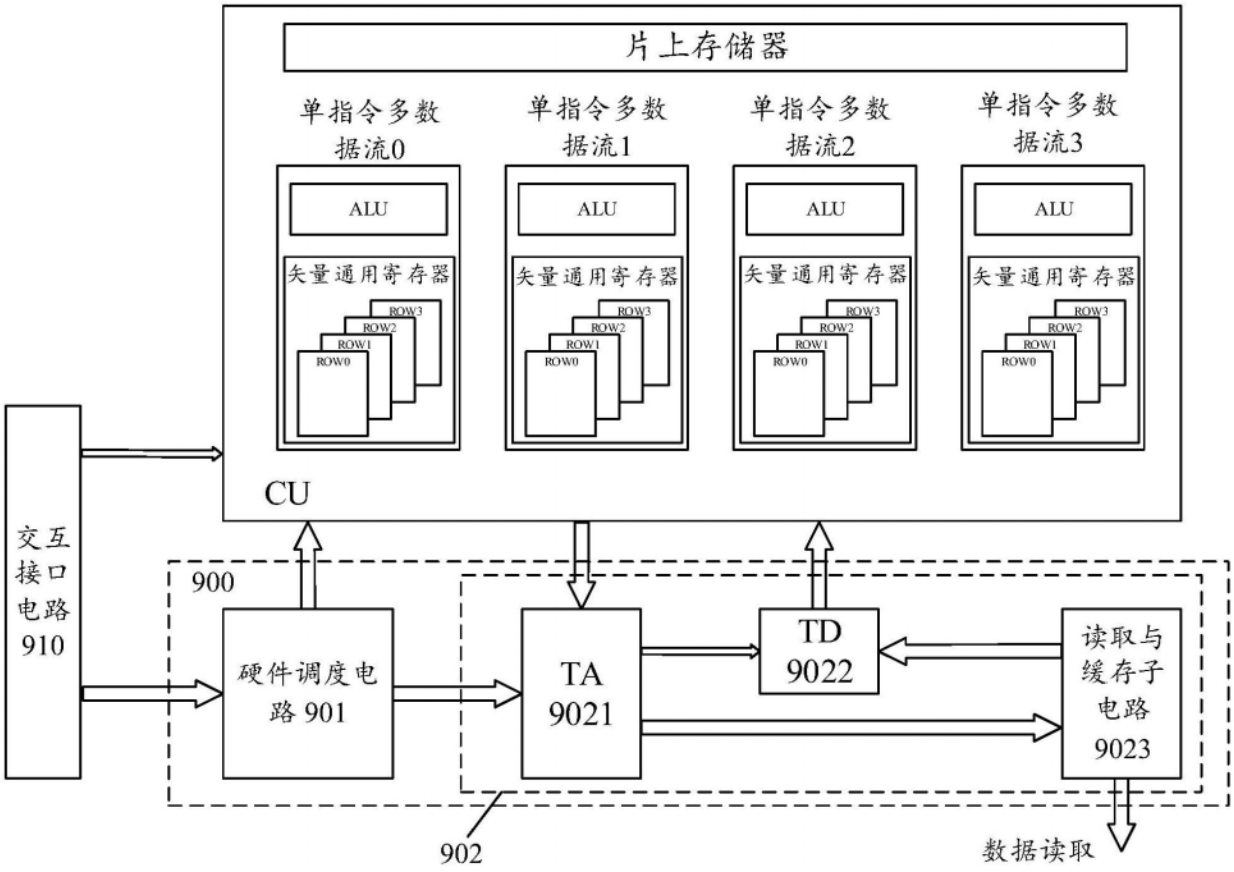

片上存储器
单指令多数据流0
单指令多数据流1
单指令多数据流2
单指令多数据流3
ALU
矢量通用寄存器
ROW3
ROW2
ROW1
ROW0
CU
交互接口电路910
900
硬件调度电路901
TA 9021
TD 9022
读取与缓存子电路9023
902
数据读取

图9

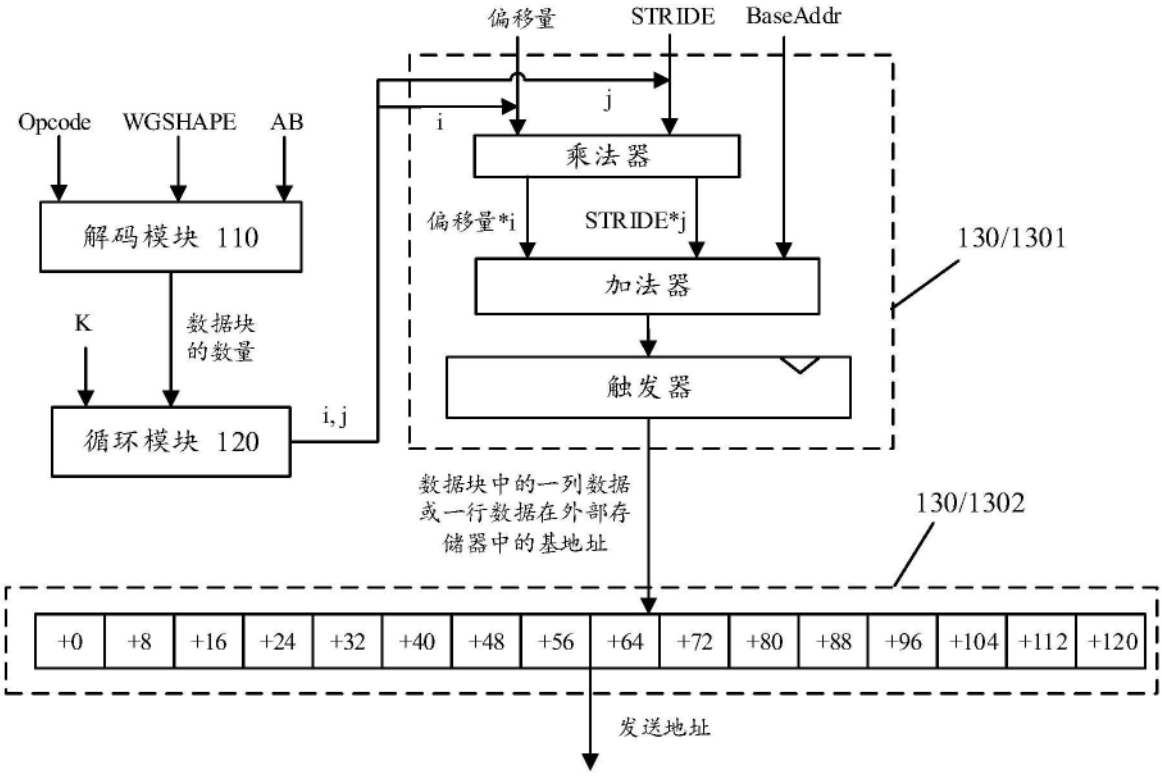

偏移量
STRIDE
BaseAddr
j
i
Opcode
WGSHAPE
AB
乘法器
偏移量*i
STRIDE*j
130/1301
解码模块 110
加法器
K
数据块
的数量
触发器
i, j
循环模块 120
数据块中的一列数据
或一行数据在外部存
储器中的基地址
130/1302
+0
+8
+16
+24
+32
+40
+48
+56
+64
+72
+80
+88
+96
+104
+112
+120
发送地址

图10