2020-2021春夏学期《计算机组成与系统结构》大作业

本作业的目的是为了更好的了解RISC-V处理器的基本架构，并把本学期所学知识用于实际的应用之中。本项作业时开放性题目，可以根据自己的知识，提出一定的创新，完成矩阵与矩阵的乘法运算。本作业会给出一定的引导性思路，可以基于这些引导性思路进行拓展并完成整个设计。

1. 任务要求

实现两个矩阵的乘法运算，两个矩阵的大小为M\*K和K\*N，其中M>1024，N，K不超过256。输入矩阵A和B的数据位宽都为8b，输出矩阵C的数据位宽为16b。由于完成乘加运算将超出16b的表示范围，因此需要对数据进行量化和更高效的表示。该乘加过程可以用ALU、PE阵列、存算电路等来实现。完成设计后需要对方案进行评估，评估指标为完成该任务所执行的周期数。

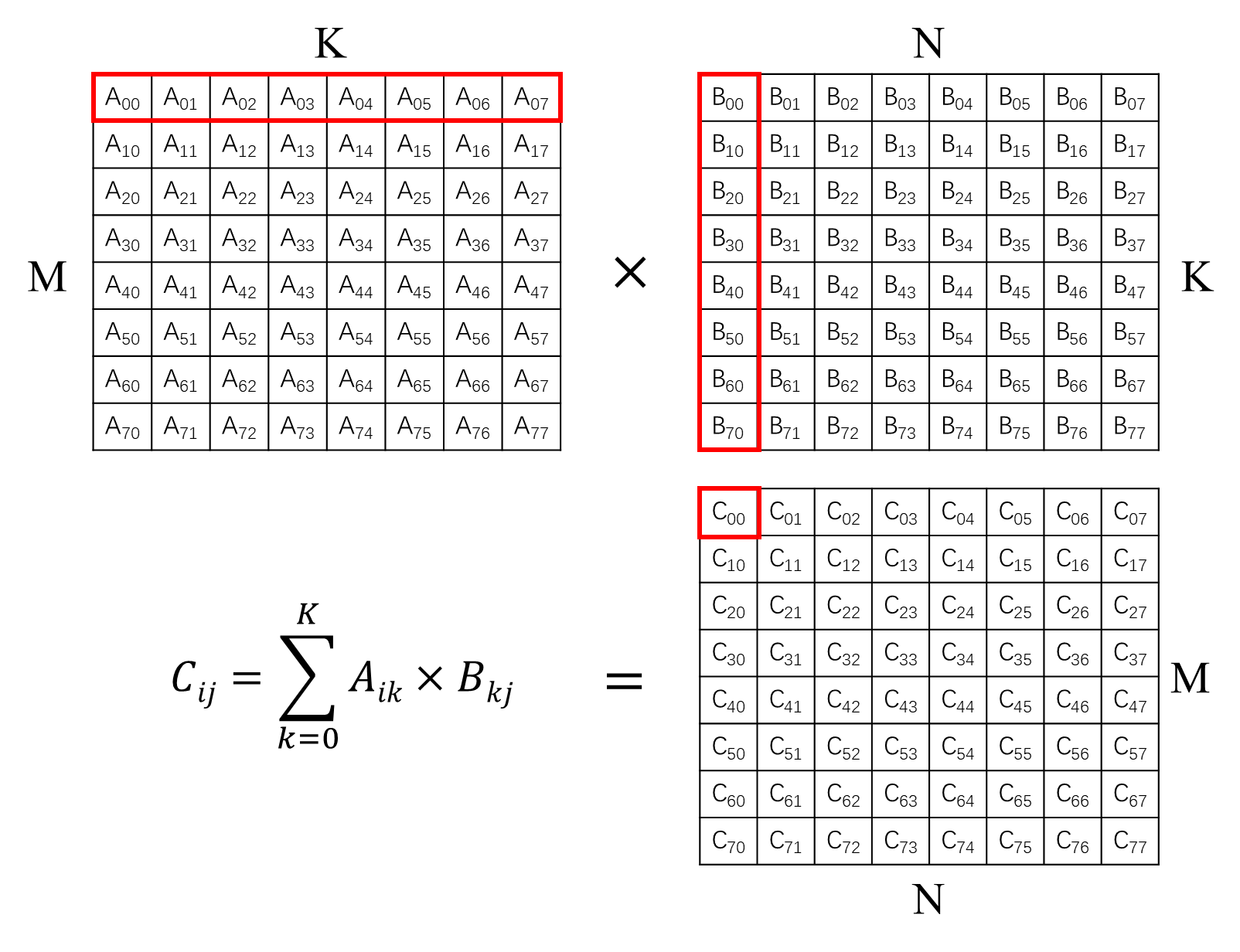


图1 矩阵乘法运算示意图

1. 开发工具

risc-v基础的工具：<https://github.com/riscv/riscv-tools>

risc-v soc开发工具：<https://github.com/ucb-bar/chipyard>

其他开发工具：<https://www.riscv-simulator.com/>

risc-v实验教程：<https://github.com/ucb-bar/riscv-sodor>

1. 设计语言

可以选择自己熟悉的程序设计语言，比如Verilog，SystemC，Chisel等。建议使用Chisel进行处理器架构性的描写。Chisel的基本教程已经上传到学在浙大课程章节目录下面。对于验证的程序（两个矩阵相乘的运算），可以使用C语言，用gcc进行编译。

1. 设计思路

这里仅提供一些用于参考的设计思路，具体的设计思路需要进一步细化。主要设计针对矩阵乘法运算的加速器电路，使用RISC-V来作为控制器或者对RISC-V进行改进。

在设计之前，先要对数据进行更加高效的量化表示。整型的计算效率高，但是数据表示范围小。而浮点型表示范围大，但是计算需要分成两部分，导致计算效率低。而且使用存算等方案来设计的化，浮点数很难实现大规模的并行计算。因为建议使用伪浮点数的表达形式，即对所有的数据使用相同的量化因子进行量化，把数据限定在-1和1之间，即第一位表示符号位，后续位表示为0-1之间对的数。在完成数据表示之后，需要考虑有符号的乘法运算的实现方案。

1）使用单核RISC-V来计算矩阵乘法运算（推荐）

该设计主要使用课堂学习的知识来设计RISC-V处理器。设计有符号乘法器来提升计算速度，使用流水线来提升计算性能，使用分枝预测来提升for循环的判断速度，使用合理的缓存大小来减少数据搬移，提升命中率。

2）使用多核RISC-V来计算矩阵乘法运算

该设计主要使用课堂学习的知识来设计RISC-V处理器。对于矩阵乘法，若使用单个ALU则需要对整个矩阵乘法运算进行多次for循环，计算效率较低。因此推荐使用多核并行计算的方法。在使用多核并行计算的时候，需要考虑循环展开、缓存的一致性、分枝预测等方法。此外也要设计针对上述数据表示的乘法器。

3）精简RISC-V指令，定制数据通路和额外指令，来为DLA量体裁衣（推荐）

由于RISC-V是相对通用的处理器，很多指令在执行该任务时不会被用到。对指令集进行裁剪可以优化数据通路，减少硬件资源，提升计算速度。此外，考虑到并行计算可以大幅度提升矩阵相乘的速度，因此可以对功能单元进行优化，比如设计一个乘加阵列来大幅度提升计算的性能。在设计乘加阵列的时候，可以结合高效的数据流，比如脉动矩阵，来减少数据的搬移。此外，由于使用了新的功能模块，建议设计新的指令并修改数据通路和控制模块，来提升计算的性能。

由于乘加阵列数量固定，那么对大的矩阵就需要进行切分，把计算的部分和存入缓存或者内存再在下一次读入完成部分和相加，直到完成全部的计算。矩阵的切割和数据流需要针对所设计的架构进行优化，从而达到最高的效率。

4）设计全新数据通路来深度融合存算阵列（控制计算一体，推荐）

首先介绍下存算阵列。存内计算的本质是向量矩阵的乘法运算阵列。为了简化本项目的设计，我们将采用如图2所示的结构。矩阵A从左边按行串行输入，一行中的每一个数据与矩阵B的每一行进行相乘，其部分和经过相加得到一行C并行输出。

使用存内计算需要先完成两项任务，首先是确定量化的表示方式，以及乘累加计算方式。如果全部都是正值，其计算过程比较简单。考虑到负值的情况，往往需要使用补码。如何处理符号位需要先进行明确。其次需要对存算阵列的计算行为建立模型，由于最后的仿真。整个计算过程可以考虑为X个时钟周期内完成一次的矩阵向量乘加运算，那么M次输入就需要M\*X个时钟周期，其中X为数据位宽。

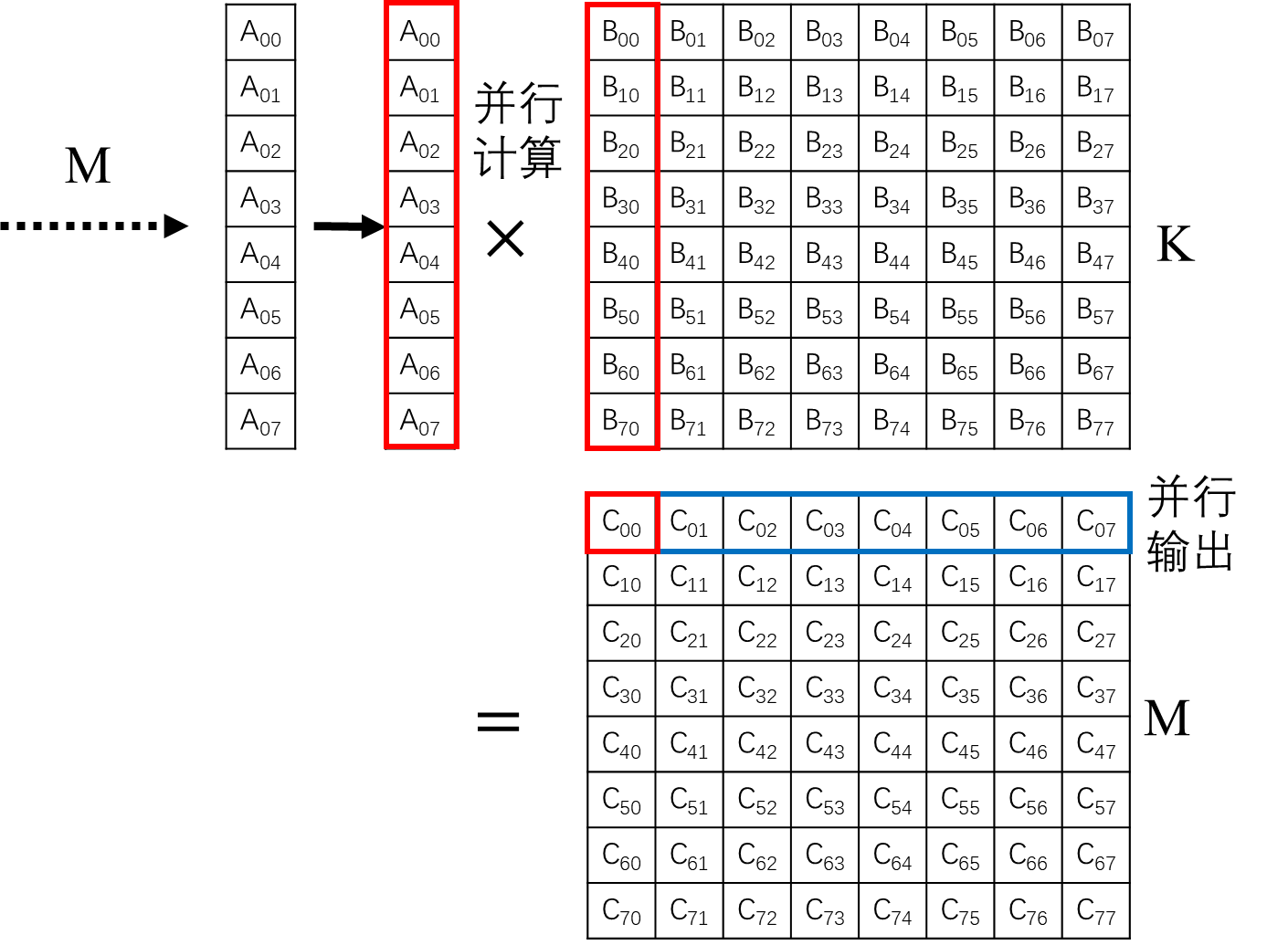


图2 存内计算示意图

该题目与3）类似，对较大的输入矩阵需要进行切割，并优化数据流。最大的区别是把3）的乘加运算阵列替换成存算阵列。与3）的主要区别是，3）的两个输入可以在乘加阵列间移动，而4）一般一个输入固定（比如B固定），另外一个输入只需要进入存算阵列，不需要在阵列内部移动。需要研究存算阵列的大小，缓存的大小对整体性能的影响。此外需要优化数据位宽以提升并行度。

5）使用RISC-V和NVDLA来搭建深度学习加速器

该方案已有较多的前期工作，比如github上的官方代码<https://github.com/nvdla/>。在使用该方案的时候，需要根据任务做一定的优化，并进行一定的对比实验。由于该方案完成度较高，对整体硬件需要了解的知识点较多，而且对深入了解RISC-V处理器的作用较小，所以个人不是太建议采用该方案。

6）结合RISC-V与存算阵列来设计存算处理器（ RISC-V作为控制器）

该方案是5）的升级版，使用存算阵列来取代NVDLA来提升计算的效率。由于RISC-V只是作为控制器，那么可以对RISC-V处理器做一定的定制与优化，并设计更高效率的总线来满足大规模的并行计算。

1. 验证手段

验证主要包括基础性测试和任务性测试。基础性测试主要是测试新增的指令、CPI、流水线冲突性测试、缓存命中率测试、分枝预测命中率测试等。任务性测试主要是对矩阵相乘的任务进行测试，测试其完成需要的时钟周期数、计算准确性，以及其他一些相关测试。此外还要对不同大小的矩阵进行测试和对比。

1. 时间节点

建议每个月递交一份设计文档，最后递交完成的设计文档、设计代码、测试代码、测试结果等。第一次递交整体项目设计方案，第二次递交原理性论证，第三次递交初步设计与测试结果，第四次递交整体系统设计方案、测试结果、系统的分析与仿真结果，建议增加对比结果。

1. 报告撰写

实验报告分成四部分，分四次递交，最后一次递交完整的版本。报告书写要清晰，有章节标题和目录，引用的内容要有参考文献。因此最后的报告需要包括：设计目标、设计内容、技术方案、创新点、实验结果、实验对比、实验分析、结论与小结，最后加上反馈意见和参考文献。

1. 评分原则

评分将根据选题难度、创新、完成度、效果、报告书写等多个方面进行综合评估，最终如果创新较好，可行性较高，但是由于设计过于复杂，只完成了主要的模块，完成度的评分占比将会降低。中间过程可以根据自己的精度做一定的调整。如果未能在规定时间内递交报告和代码，每延迟一天扣1分扣完为止。若发现抄袭将不给分。