文章编号:1671-4598(2024)05-0267-07 DOI:10.16526/j.cnki.11-4762/tp.2024.05.038 中图分类号:TP183 文献标识码:A

基于 FPGA 的深度可分离卷积加速器研究

画芊昊，李 博，杜宸罡
（中北大学 仪器科学与动态测试教育部重点实验室，太原 030051）

摘要：设计了一种基于 FPGA 的低功耗深度可分离卷积加速核；根据 PW 卷积和 DW 卷积计算中的共性，采用一种固定乘法阵列通过改变特征和权重输入数据流的方式实现两种卷积的计算结构，最大化 DSP 的利用率；针对 8 位非对称量化中符号位可能会溢出的问题，采用符号位单独处理的方法重新封装了双乘法器结构；通过层内 7 级流水结构保证每个周期数据处理的并行度；在 Zynq UltraScale+系列 FPGA 上成功部署了加速结构；经实验测试，提出的加速结构在提高网络推理速度的同时降低了片上资源的依赖度和整体功耗，原生 MobilenetV2 在所提 FPGA 加速器上的平均吞吐率高达 130.6 GOPS 且整体功耗只有 4.1 W，满足实时边缘计算的要求；相比其他硬件平台，能效比有明显提升；与 FPGA 上的同类型加速器相比，在性能密度（GOPS/LUT）、功率效率（GOPS/W）和 DSP 效率（GOPS/DSP）上均有优势。

关键词：FPGA；硬件加速器；卷积神经网络；非对称量化；Mobilenet

Research on Deep Separable Convolution Accelerator Based On FPGA

HUA Qianhao，LI Bo，DU Chengang
（Key Laboratory of Instrumental Science and Dynamic Testing，Ministry of Education，
North University of China，Taiyuan 030051，China）

Abstract：A low power deep separable convolution accelerator kernel based on FPGA is designed. According to the commonality of Pointwise (PW) convolution and Depthwise (DW) convolution calculations，the fixed multiplicative array is used to realize the two convolution calculation structures by changing the feature and weight input data stream，so as to maximize the utilization of DSP. In order to solve the problem that the sign bit may overflow in the 8-bit asymmetric quantization，the double multiplier structure is repackaged by using the sign bit processing method. The parallelism of data processing in each cycle is guaranteed by the 7-level pipelining structure in the layer. The accelerator structure is successfully deployed on the Zynq UltraScale+ series FPGA；Through the experimental test，the results show that the proposed acceleration structure can improve the inference speed of network and reduce the dependence of on-chip resources and overall power consumption. The average throughput of the original MobilenetV2 on the proposed FPGA accelerator is as high as 130.6 GOPS，and the overall power consumption is only 4.1 w，which meets the requirements of real-time edge computing. Compared with other hardware platforms，the energy efficiency ratio is significantly improved；Compared with the same type of accelerator on the FPGA，it has advantages of performance density (GOPS/LUT)，power efficiency (GOPS/W) and DSP efficiency (GOPS/DSP).

Keywords：FPGA；hardware accelerator；convolutional neural network (CNN)；asymmetric quantization；Mobilenet

0 引言

近年来深度学习在人工智能领域的快速发展，使得 AI 算法逐渐融合于各个领域并展示出了对于复杂任务非凡的处理能力和准确性。其中卷积神经网络（CNN，convolutional neural network）因其出色的图像特征提取能力，在机器视觉领域引起了极大关注[1]。随着网络的不断发展，其精度在不断提升的同时带来的是不断多样化的卷积算子和越来越庞大的网络规模。文献［2］提出的 GoogleNet 中使用大量 1×1 卷积来对特征进行降维处理，并将一个 $n\times n$ 卷积拆解为一个 $n\times 1$ 和 $1\times n$ 卷积来降低网络计算量；文献［3］提出一种新的可变形卷积，在卷积过程中引入偏移量，使得感受野不再是固定的形状，可以更好地贴近物体的实际轮廓；文献［4］在 Mobilenet 网络结构的基础上提出 G-bottleneck 块，在部分原始特征的基础上利用一系列计算代价较小的线性变化来生成其余所需的“幻影”特整图；文献［5］在 2016 年 CVPR（Conference on Computer Vision and Pattern Recognition）会议上提出了一种残差结构，通过融合浅层网络和深层网络不同维度的特征，成功缓解了网络退化现象，这使得卷积神经网络的规模可以往更深层发展，同年残差神经网络 ResNet50 的参数量达到了 25 M，计算量更是达到了惊人的 3.8×10^9 FLOPS。

越来越多变的卷积算子和日益膨胀的网络规模，使得普通的嵌入式 CPU 的计算能力和内存带宽无法满足实时性的要求，因此卷积神经网络的硬件加速开始备受关注。图

收稿日期：2024-02-03； **修回日期**：2023-03-25。
作者简介：画芊昊(1998-)，男，硕士生。
李 博(1972-)，男，博士，副教授。
引用格式：画芊昊，李 博，杜宸罡. 基于 FPGA 的深度可分离卷积加速器研究[J]. 计算机测量与控制，2024，32(5)：267-273.

形处理单元（GPU，graphics processing unit）和专用集成电路（ASIC，application-specific integrated circuit）以及现场可编程门阵列（FPGA，field programmable gate array）一直是加速器设计的主流平台，然而 GPU 的高功耗使其在边缘计算场景下的表现并不太理想。ASIC 硬件加速器可以通过定制化内存层次结构和分配专用运算资源的方式在较低的功耗下实现不错的计算性能，但它的开发成本高、设计周期长且灵活性一般需要有针对性地调整输入数据流才能充分发挥其高效的运算能力。作为一种替代方案，基于 FPGA 的加速器具有高度可定制化和功耗低的优点可以在相对有限的内存、I/O 带宽和逻辑资源下以合理的成本提供较高的加速性能[6-7]。如何在 FPGA 平台上提高资源利用率设计一款高能效比的加速结构就很有工程意义。

CNN 加速器设计核心是最大化结构利用率和数据流优化，宗旨是提高性能密度平衡并行度与资源使用率。对于数据流的优化，文献［8］提出一种全流水的加速思路，该团队将整个网络放置到 FPGA 上进行流水处理，网络各层之间同时运算，理论上可以发挥出板卡的全部性能。但 CNN 的层间数据依赖会随着深度的提升逐渐增强，这就使得部署层数较深的网络时，前期的模型评估和超参数调优会非常复杂，各层的资源消耗难以平衡；文献［9］总结 Mobilenet 运算中的特点，提出一种半跨层流水的结构，通过平衡点卷积（PW，point wise）和深度卷积（DW，depth wise）的并行度，使 DW 不用完全等待 PW 的全部结果再开始运算，而是将一次 DW 运算重叠在其上下两层的 PW 运算期间来实现跨层流水。但 Mobilenet 中 DW 卷积只占整体计算量的 3%，采用这种优化形式复杂化了整体数据流且速度提升有限。文献［10］采用的是传统的层内流水结构，通过优化每个层内的数据流实现每个时钟内对计算资源的最大化利用。对于最大化结构利用率，文献［11］对于 PW 和 DW 卷积均使用两套 PE（Processing Element）阵列，这就使得在进行一类卷积期间，用于实现另一类卷积的运算资源是没有被完全利用的；文献［10］复用了池化和卷积操作的加法树；文献［12］使 GhostNet 中的传统卷积和 DW 卷积共享了一套 PE 阵列，但传统卷积和 DW 卷积在滑动窗的设计上本身就具有一定的相似性，因此这种复用方法不能完全适用于其他算子间计算逻辑相差较大的网络，同时文献中对整体的并行度采用的是 $P\times R\times Q$ 的形式，其中 P 和 R 分别表示基础乘法阵列中 PE 的个数和滑动窗的个数，前者在卷积核已知的情况下是固定的（3×3 卷积时固定为 9），后者则不能定的很大（滑动窗会消耗大量移位寄存器），只有 Q（同时处理特征图的个数）可以调动，但在网络较浅时特征通道数较少，过大的 Q 值会使浅层运算时浪费部分计算资源。

针对以上两个问题，本文设计了一种面向轻量级网络 Mobilenet 的深度可分离卷积算子通用加速核。（1）根据 PW 卷积和 DW 卷积计算中的共性，设计了一种固定乘法阵列通过改变特征和权重输入数据流的方式实现两种卷积的计算结构，最大化了乘法器的利用率；（2）针对 8 bit 非对称量化中符号位可能会溢出的问题，采用符号位单独处理的方法重新设计了双乘法器结构；（3）优化层内数据流，通过层内 7 级流水结构保证每个周期计算阵列的全功率运行，同时复用 PW 卷积的累加模块用于倒残差的计算。最终，在 Zynq UltraScale+系列 FPGA 上成功部署了加速结构，平均吞吐率达到 130.6 GOPS 且整体功耗只有 4.1 W。

1 深度可分离卷积算法分析

1.1 卷积神经网络

卷积层是卷积神经网络（CNN）最重要的组成部分，卷积计算占据网络整个计算量的绝大部分。相比于传统的全连接层，卷积层具有局部连接和参数共享的优势。一个卷积核就相当于一个数字滤波器，其天然就具有数字图像特征提取的属性，网络将卷积核的权重值当作待训练的参数，通过损失函数的约束并采用反向传播和梯度下降方法更新参数，从而训练网络对于某些任务需要信息的特征提取能力。这使得卷积神经网络更加适合处理图像任务。一次完整的 CNN 运算一般包括卷积层、BN 层、非线性激活层和池化层。其中卷积层用于特征提取运算原理如图 1 所示；BN 层的表达式为：

$$y_i=\frac{x_i-\mu}{\sqrt{\sigma^2+\varepsilon}}\times\gamma+\beta \tag{1}$$

其中：β 和 γ 是可训练参数，μ 和 σ^2 是输入的均值和方差，ε 为一小值用于防止分母为零。数据在进入下一层前通过 BN 层的归一化处理可以加速网络收敛速度并防止训练过程中的梯度消失[13]；激活层增强网络的表达能力使网络具备处理复杂问题的可能性；池化层一般用于下采样和防止过拟合。

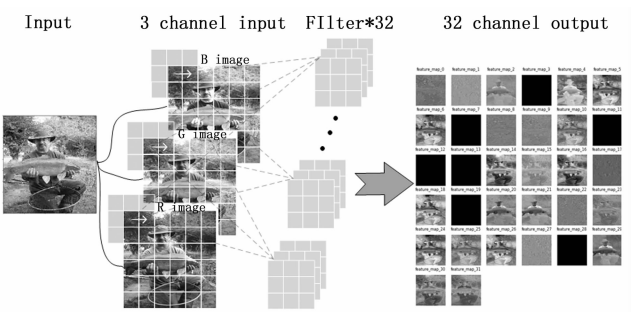

图 1 卷积层特征提取过程

1.2 深度可分离卷积

深度可分离卷积作为因式分解卷积的一种，因其在 MobileNet 模型中成功应用而广为熟知。它的核心思想是将传统卷积分解为一步 Depthwise convolution（DW）和一步 1×1 卷积（PW），这么做可以降低网络的参数和计算量，使其可以更容易在嵌入式平台部署：

$$D_K\times D_K\times M\times N\times D_F\times D_F \tag{2}$$

$$D_K\times D_K\times M\times D_F\times D_F+M\times N\times D_F\times D_F \tag{3}$$

$$\frac{D_K\times D_K\times M\times D_F\times D_F+M\times N\times D_F\times D_F}{D_K\times D_K\times M\times N\times D_F\times D_F}=\frac{1}{N}+\frac{1}{D_K^2} \tag{4}$$

图 2 为深度可分离卷积图示，其中 DW 卷积核只负责对一个通道中的特征图进行卷积，可以通过设定步长控制输出特征图的尺寸，PW 卷积负责通道融合并决定输出特征图的通道数。公式（2）为标准卷积的计算量，其中 D_K 和 D_F 为卷积核和输出特征的尺寸，M 是特征深度 N 是通道数。公式（3）为深度可分离卷积的计算量，公式（4）是二者比值。可以看出当 $D_K=3$ 时，使用深度可分离卷积代替标准卷积可以降低 7 到 8 倍的计算量（$N \geqq 32$）[14]。

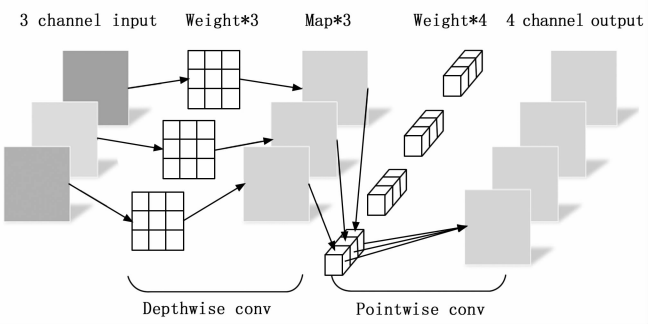

图 2　深度可分离卷积计算过程

输入特征通过 13 组深度可分离卷积后的结果通过平均池化、一层全连接和 Softmax 输出 Imagenet 数据集下的 1 000分类结果。其中 PW 卷积占整个网络计算量的 94%是加速任务重中之重。

2　非对称量化

理论上通过扩大卷积神经网络的参数规模，网络可以近似拟合任意复杂度的目标函数[15]。但代价是越来越庞大的网络参数，如何压缩模型体积降低网络对内存和带宽的依赖使其更容易在嵌入式硬件平台上部署，是软硬协同加速器设计要解决的第一个问题。

常见的模型压缩方法包括模型剪枝、知识蒸馏、神经网络结构搜索算法和模型量化。其中前 3 种方法都是针对网络结构的优化[15-17]。而模型量化则是直接针对参数本身的优化，通过将高位宽浮点数映射到低位宽的整数域中，可以在不改变模型结构的前提下直接实现模型压缩和全整数推理的目的，公式（5）为映射关系，q 是实数 r 在整数域的象，Z 为偏置 S 为量化因子[18]。

$$r = S \times (q - Z) \tag{5}$$

量化按照 0 点映射的位置可以分为对称量化和非对称量化，对称量化的方式一般要求被量化的数值分布是对称的，且输出存在负值的激活函数。非对称量化则不受分布和激活的影响[19]，通过引入量化点偏移 Z 来调整缩放后的数值，其表示范围相对对称量化会更加准确。使用 Imagenet 数据集在 tensorflow 框架下对 MobilenetV2 模型采用 uint8 非对称量化感知训练，量化后的模型相对原生模型的准确率仅下降了 0.3%，精度损失基本可以忽略。

3　加速器设计

3.1　加速器总体结构

通过研究 PW 卷积和 DW 卷积计算中的共性，创新性地提出了一种固定乘法阵列通过改变特征和权重输入数据流的方式实现两种卷积的计算结构，提高了 DSP 的利用率。针对 8 bit 非对称量化中符号位可能会溢出的问题，采用符号位单独处理的方法重新封装了双乘法器结构。优化了层内数据流，通过层内 7 级流水结构保证每个周期计算阵列的全功率运行。同时复用 PW 卷积的累加模块用于倒残差结构的计算。

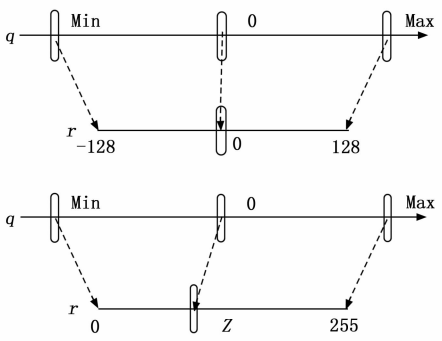

图 3　对称量化与非对称量化映射关系

加速器整体由四路 DMA（两路 Weight 通道，一路 Feature 通道）、任务调度中心（Dispatching center）、数据处理核心（Data Processing Unit）和控制寄存器组成，如图 4 所示。

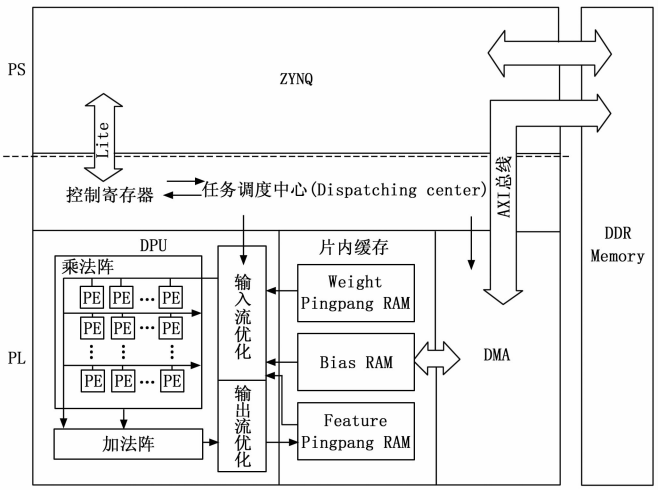

图 4　加速器整体结构

3.1.1　PS 与 PL 间数据搬运

DMA 负责 PL 与 PS 端 DDR 的数据交互，为了提高灵活度和精细化资源使用率，本文没有直接调用 IP 核，而是自主设计了这个模块。它主要由 3 个部分组成：AXI 协议层实现逻辑、位宽转换逻辑和 RAM 块接口控制逻辑。以内嵌位宽为 1 024 的 RAM 为例，AXI 协议层实现逻辑负责与 AXL Slave 端完成握手协议，将待读写数据搬运至 AXI 总线；位宽转换逻辑在 AXI 读时，将数据以 64 位宽从给定起始地址顺序写入 RAM（设为 64 为了同时适配 ZYNQ7 和 UltraScale+）。在 AXI 写时，位宽转换逻辑从给定的 RAM

地址读取数据后拆分成数个 64 位数据顺序发送给 AXI 协议逻辑；RAM 块接口控制逻辑负责切换 RAM 端口的连接，在非 AXI 读写期间释放 RAM 端口，使其可以用于片内缓存。

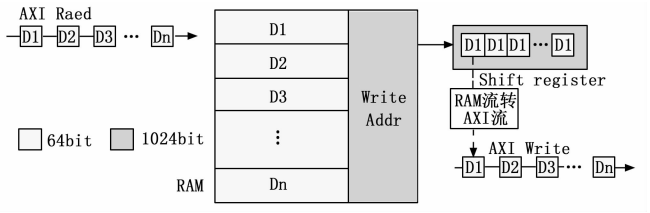

图 5 DMA 数据搬运逻辑

3.1.2 调度中心

任务调度中心是加速器顶层的核心状态机。它从控制寄存器中读取网络的结构信息和 PS 端的控制信息，将命令发送给 DPU 网络推理。收到 DPU 的反馈信息后转换状态进行网络下一块的运算，直至一次任务完成。

3.1.3 DPU 流水线设计

数据处理核心（DPU）是加速器中的核心模块，它基本互联了片上的所有模块。它由输入数据流优化器、核心乘法阵、加法阵和输出数据流优化器组成。乘法阵和加法阵联合实现卷积运算，输出数据流优化器对结果进行量化操作将最终结果写入片内缓存。如图 6，该模块采用流水线设计，收到任务调度中心的命令后，输入数据流、卷积运算和输出数据流不间断地流水操作，完成计算后返回标志位准备开始下轮运算。

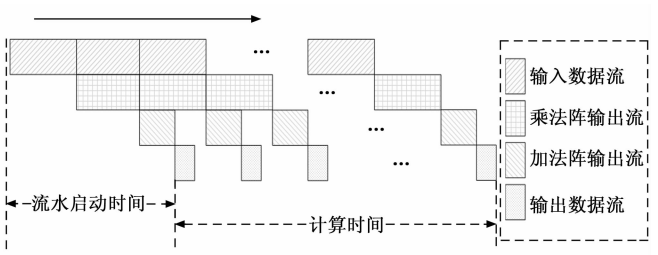

图 6 DPU 总体数据流水

3.2 乘法阵设计

乘法阵列总体为高并行度的 DSP 阵列，对输入数据流没有要求只是单纯将乘数端口与被乘数端口的数据对应相乘，并并行输出乘积结果。输入数据格式为 8 bit，乘积结果暂存为 32 bit 用于后续累加和缩放。

3.2.1 非对称量化下双乘法器的优化

FPGA 上的 DSP48E1 最大可已进行一 25×18 位的乘法，UltraScale+系列使用的 DSP48E2 将位宽升级到了 27×18。若每个 DSP 只进行一次 8 位乘法，就浪费了大量的位宽。通过将两个 8 位数据按一定方法拼接成 $\{A,'\ d_0,\ B\}$，可以实现从 $\{A,'\ d_0,\ B\}$ xC 的结果中完全分离 AxC 和 BxC 的结果[20]，实现在一个 DSP 中进行两次 8 bit 乘法（如图 7 所示）。

由于模型使用 Relu 激活，采用 int8 量化将浪费一定量

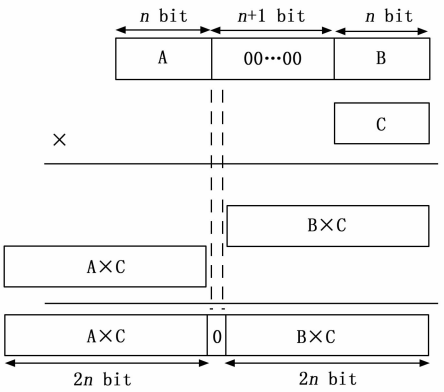

图 7 双乘法器计算过程

化精度（激活的结果只会被量化到 0～127，浪费掉 1 bit 精度），固本文采用 uint8 非对称量化策略。在量化感知训练中会对 BN 层和激活层（Relu）进行融合。BN 运算可以通过作用于权重将其体现在融合后新的权重值本身，激活函数则是通过量化公式体现在偏移量 Z（0 所对应的量化值）和量化因子 q 中。

设 r_{out}（i，j）表示输出的第（i，j）个元素，$r_n^{(i,j)}$ 表示特征第（i，j）个元素及其周围 8 个元素所组成的集合中第 n 个元素，r_n^W 表示卷积核 W 中第 n 个元素，r_b 表示偏置。则 DW 卷积的推理公式为：

$$r_{\text{out}}(i,j)=\sum_{n=0}^{8} r_n^{(i,j)}\times r_n^W+r_b \tag{6}$$

将量化的映射公式（5）带入可推得量化推理公式为：

$$S_{\text{out}}(q_{\text{out}}(i,j)-Z_4)=\sum_{n=0}^{8} S_1(q_n^{(i,j)}-Z_1)\times S_2(q_n^W-Z_2)+$$

$$S_3(q_b-Z_3)\Rightarrow q_{\text{out}}(i,j)=\frac{S_1S_2}{S_{\text{out}}}\sum_{n=0}^{8}\sum_{n=0}^{8}(q_n^{(i,j)}-Z_1)$$

$$(q_n^W-Z_2)+\frac{S_3}{S_{\text{out}}}(q_b-Z_3)+Z_4 \tag{7}$$

在 C++量化推理验证中发现，量化后的 uint8 数据在减去偏移 Z 后其结果并不能完全用一个 int8 表示，每层推理中都会有 10%左右的数据发生溢出，所以偏移无法在数据预处理中消除，这就需要对双乘法器进行重新封装。针对 DSPE1 和 DSPE2 可以采用两种封装方法：（1）首先待操作数减去偏置得到 3 个 9 bit 数，其中最高位为符号位，低 8 位为结果的绝对值。接着将 3 个 uint8 数据送入双乘法器中运算得到两个 16 位结果。最后根据 3 个操作数的符号位将两个 16 位结果扩充致 int32。除去减去偏移和位数扩充两步操作，实现双 int9 乘法共计引入 1 次移位、4 次判断和两次取反操作，且可以同时适配 DSP48E1 和 DSP48E2。（2）使用有符号乘法器 IP 核，位宽设为 27×9。如图 8 所示，将 a 和 b 拼接为 $\{a,\ 9'\ d_0,\ b\}$ 与 C 相乘得到 36 位结果 O，其中 $out1=(a\times C)=(C\ [8]==0)?\ O\ [35:\ 18]:\ O\ [35:\ 18]+1$，$out2=(b\times C)=(b\ [8]==0)?\ O\ [17:\ 0]:\ O\ [17:\ 0]-(C<<9)$。额外引入两次判断一次加减，只适用于 DSP48E2。

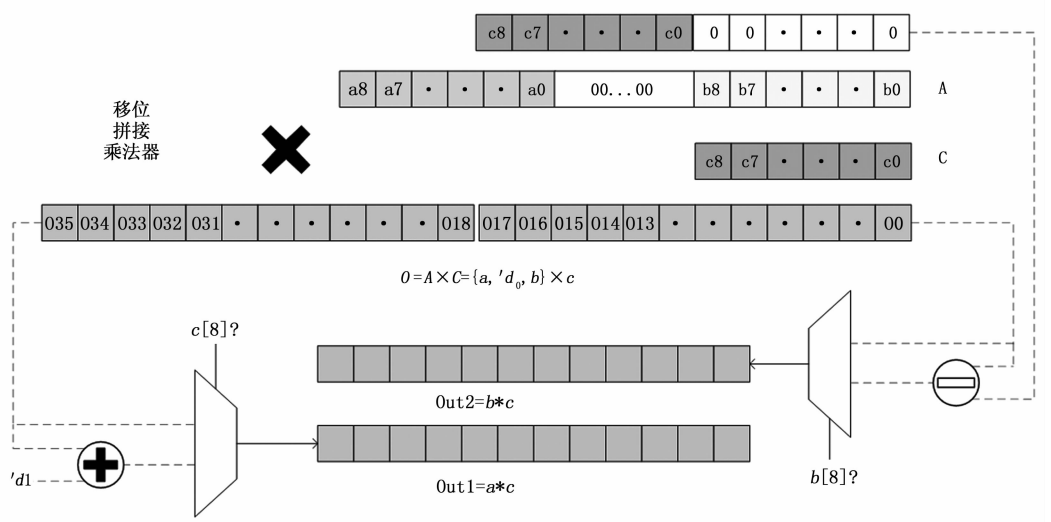

图 8　双乘法器封装（2）

3.2.2　时序优化设计

由于整个乘法器阵列采用 DSP 大并行结构，时钟频率提高后会对综合后的布局布线产生很大压力。为了使本结构可以适配更多资源总量不同的硬件平台，本身使用 DSP 内部自带的触发器下，在输入和输出也要穿插一到两级触发器方便综合器的布局操作。对于高并行度产生的大扇出问题，在 RTL 设计时采用信号复制的驱动方式（加上限制防止被综合优化），约束时需要将一些控制信号加入 BUFFG 中增加驱动能力降低 Net delay。

3.3　输入数据流优化器

输入数据流分为 Weight 数据流、Feature 数据流和 Bias 数据流，每个数据流又分为 DW 模式和 PW 模式，总计设计六股数据流。如图 9 所示，Weight 以卷积核为单位存储，Feature 按通道一幅幅存储。

表 1　符号定义

名称	符号含义
H	特征图高度
W	特征图宽度
C	特征图深度
K	卷积核尺寸
TC	卷积核通道数

在 DW 计算期间，Weight 与 Feature 只需顺序读写。输入特征图一次读取大于四行的数据进入滑动窗口开始运算，当滑动窗口的并行度设置为 *n* 时，滑动窗输出元素周围九宫格的数据（边缘给 0）并往后滑动 *n* 个元素。当行计数器计数完毕，重新往滑动窗填充一行数据。当页计数器计算完毕后，重新读取一组 Weight（新的卷积核）。滑动窗输出的 *n* 个九宫格个数据会和当前通道的卷积核一起送入双乘法阵列进行运算。

在 PW 计算期间，Weight 与 Feature 会有回读的需要。以 Feature RAM 的位宽为 1 024（128 字节）为例，输入特征图的尺寸是 2 的整次幂时，则网络前 *M* 层的特征图都是可以被 128 整除的（比如输入为 128×128 时网络从第 23 层开始被下采样为 8×8）。设第 *L* 层（*L*＜*M*）的输入特征图为 *H*×*W*×*C*（*H*＝*W*＞32），*T* ＝ *H*×*W*/128（∈*Z*），则

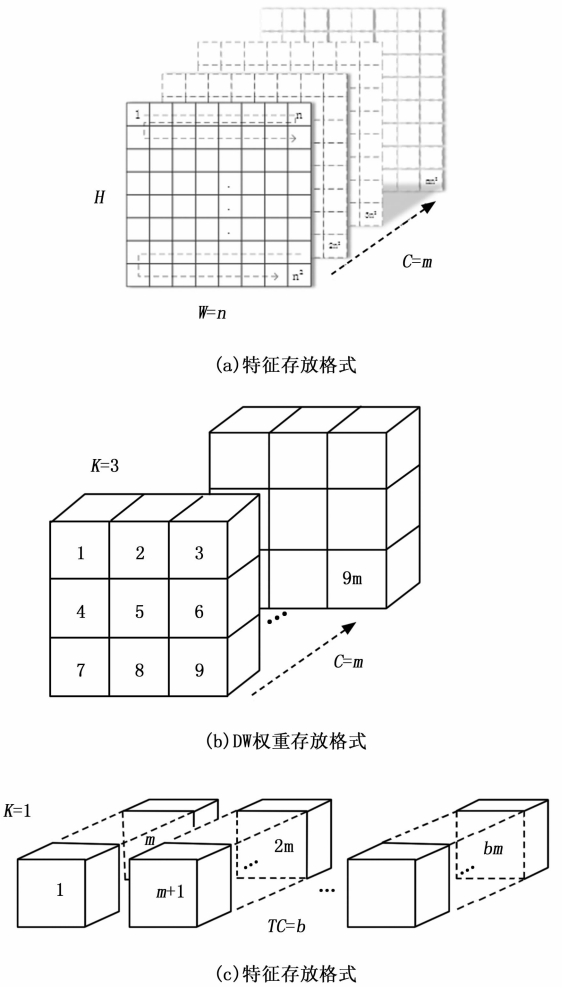

图 9　数据存放格式

分别读取 Feature RAM 的第 1、（T+1）、（2T+1）……（（C−1）T+1）个数据和 Weight RAM 中第一个卷积核的第 1，2，3，…，C 个数据送入乘法阵和 PW 加法阵中，输出第一幅结果中的前 128 个元素。第二轮，Feature 流中数据地址加 1，Weight 流数据不变，输出第一幅结果中前 256 个元素。以此类推，经过 T 轮运算后可得到第一幅结果的所有元素。当 $L>M$ 时，此时特征图虽不能被 128 整除，但是 128 的因数。Feature 流无需再跳跃读取，顺序读取 V（此时 128＝$H\times W\times T$）轮后输出 TC 幅结果的所有元素。

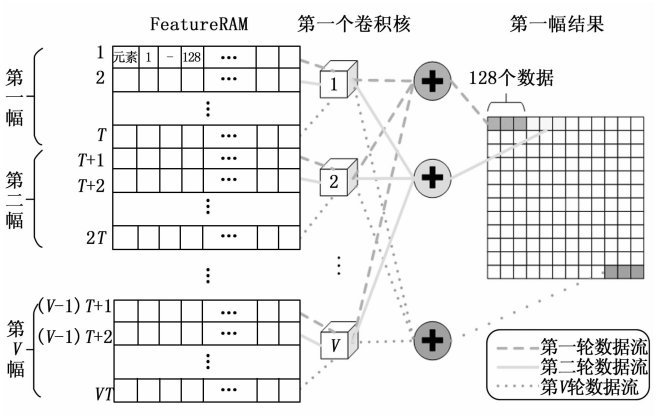

图 10 PW 期间数据流

4 实验

本节将从多个方面对加速器的整体性能进行测试。通过横向对比当前其他同样先进的加速器设计，来评估本文加速器的优势和应用价值。通过纵向对比不同硬件平台的网络推理性能来评估加速器在实际应用中的有效性。

4.1 实验设计

1）加速器的整套开发在 Windows 环境下进行。在 Tensorflow 框架下进行网络的搭建、训练和量化。加速器的所有结构在 Vivado 2018.3 中使用 VerilogHDL 进行设计。硬件平台为 Xilinx 的 ZYNQ 系列 MPSoC（XCZU9EG FPGA＋ARM Cortex-A53）。

2）本实验加速的网络结构为 MobilenetV2。将待处理数据集存储在 SD 卡中，上电后由 ARM 将其读进 DDR（4 GB），加速器部分通过 AXI 总线接收 ARM 指令和从 DRR 中读取图片数据与网络参数至片上缓存并返回推理结果。

3）由于不同硬件平台间的资源和功耗存在较大差异，本文将主要从吞吐量（GOPs），能效比（GOPs/w）和性能密度[21]（GOPs/LUT 和 GOPs/DSP）三个指标来验证加速器性能。

4.2 性能评估

4.2.1 与 CPU 与 GPU 平台的性能对比

GPU 和专用集成电路（ASIC）以及 FPGA 一直是加速器设计的主流平台，本文通过对比 MobilenetV2 在 GPU、CPU 以及加速器上的推理性能表现来验证设计在能效比上的优势。其中 CPU 使用 Intel 酷睿 i7 6700HQ 处理器，主频 2.6 GHz 功耗 45 W。GPU 平台使用的是 NVIDIA GTX960m 桌面级显卡，主频 1.1 GHz 功耗 65 W。CPU 和 GPU 均在 Tensorflow 框架下进行推理验证，采用 Imagenet 数据集测得 100 幅图片的平均推理时间如表 2 所示（输入尺寸 128 * 128 * 3）。加速器相比高性能 CPU 速度提升了 1 倍而功耗仅为其十分之一。相比于桌面级 GPU 在吞吐量上略微落后，但能效比却是它的 11 倍。证明本文实现的硬件加速器具有高性能和高能效比的优点，适合于实时性要求较高的边缘计算场景。

4.2.2 与国内外相关工作的对比

由于不同硬件平台间的资源和功耗存在较大差异，本文将主要从吞吐量（GOPs），能效比（GOPs/W）和性能密度[21]（GOPs/LUT 和 GOPs/DSP）三个指标来综合验证加速器的性能。

由表 3 和图 11 可以看出，本文所设计加速器在能效比和性能密度方面均有优势。由于本文的 PW 卷积与 DW 卷积共用乘法阵列且重新封装了双乘法器，所以 GOPs/DSP 是对照组中最高的。文献［8］和文献［9］在 GOPs/LUT 高于本文，但文献［8］采用的是 4 bit 量化位宽压力较小且网络算子单一。文献［9］则是使用更多的逻辑资源换来的高吞吐量，不适合低成本低功耗的边缘计算场景。

表 2 不同平台性能对比

类别	推理延时	功耗	时钟	吞吐量	能效比
CPU	3.17 ms	45 w	2.6 GHz	63 GOPs	1.4 GOPs/w
GPU	1.13 ms	65 w	1.1 GHz	180 GOPs	2.7 GOPs/w
本文加速器	1.53 ms	4.1 w	300 MHz	130.6 GOPs	31.8 GOPs/w

表 3 与其他相关工作的比较

	量化位宽	LUT	DSP	时钟	功耗	吞吐量	能效比(GOPs/w)
文献[11]	8 bit	58 K	244	150 Mhz	4.1 w	96.3 GOPs	23.48
文献[22]	8 bit	91 K	387	200 Mhz	4.8 w	100.3 GOPs	20.89
文献[8]	4 bit	42 K	360	166 Mhz	—	126.7 GOPs	—
文献[9]	8 bit	160 K	2 070	333 Mhz	—	485.4 GOPs	—
本文	8 bit	59 K	200(128＋72)	300 Mhz	4.1 w	130.6 GOPs	31.85

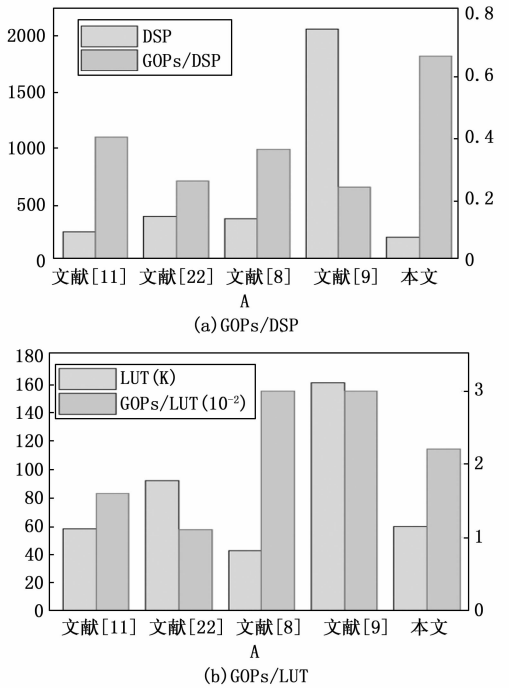

图 11　性能密度

5　结束语

本文设计了一种面向轻量级网络 Mobilenet 的深度可分离卷积加速结构。根据 PW 卷积和 DW 卷积计算中的共性，设计了一种固定乘法阵列通过改变特征和权重输入数据流的方式实现两种卷积的计算结构，最大化了乘法器的利用率；针对 8 bit 非对称量化中符号位可能会溢出的问题，采用符号位单独处理的方法重新设计了双乘法器结构；优化层内数据流，通过层内 7 级流水结构保证每个周期计算阵列的全功率运行，同时复用 PW 卷积的累加模块用于倒残差的计算。加速器最终的平均吞吐率为 130.6 GOPs，功耗仅为 4.1 W。相比于其他加速器结构，在能效比上和性能密度上均有明显提升。后续工作将优化滑动窗的设计，提高 DW 期间对乘法阵的利用率，同时寻求加速器对 MobilenetV3 的兼容性。本研究为边缘计算场景下 CNN 算法的推理加速，可以推广到各领域的机器视觉任务中，具有较高的应用推广价值。

参考文献：

[1] LECUN Y，BEVGIO Y & HINTON G，Deep learning [J]. Nature，2015，521：436－444.

[2] SZEGEDY C，VANHOUCKE V，IOFFE S，ET AL. Rethinking the inception architecture for computer vision [J]. CoRR，2015.

[3] DAI J，QI H，XIONG Y，et al. Deformable convolutional networks [J]. IEEE，2017.

[4] HAN K，WANG Y，TIAN Q，et al. GhostNet：more features from cheap operations [J]. Conference on Computer Vision and Pattern Recognition (CVPR)，2020，1577－1586.

[5] HE K，ZHANG X，REN S，et al. Deep residual learning for image recognition [J]. IEEE，2016.

[6] SHAWAHNA A，SAIT S M，EL-MALEH A. FPGA-based accelerators of deep learning networks for learning and classification：a review [J]. ACCESS，2019.

[7] GUO K Y，ZENG S L，YU J C，et al. A survey of FPGA-based neural network inference accelerators [J]. ACM Trans. 2019，Reconfigurable Technol. Syst. 12，1，Article 2 (March 2019)，26.

[8] 包振山，郭俊南，张文博，等. UltraAcc：基于 FPGA 流水架构的低功耗高性能 CNN 加速器定制设计 [J]. 计算机学报，2023，46 (6)：1139－1155.

[9] WU D，et al. A high-performance CNN processor based on FPGA for mobileNets [C] // 2019 29th International Conference on Field Programmable Logic and Applications (FPL)，Barcelona，Spain，2019：136－143.

[10] 蹇　强，张培勇，王雪洁. 一种可配置的 CNN 协加速器的 FPGA 实现方法 [J]. 电子学报，2019，47 (7)：1525－1531.

[11] QUOC P C，THINH N T. Efficient FPGA-Based convolutional neural network implementation for edge computing [J]. JAIT，2023，14 (3)：

[12] RUIHENG Y，ZHIKUN C，BIN'AN W，et al. A lightweight detection method for remote sensing images and its energy-efficient accelerator on edge devices [J]. Sensors (Basel，Switzerland)，2023，23 (14)：

[13] IOFFE S，SZEGEDY C. Batch normalization：accelerating deep network training by reducing internal covariate shift [J]. CoRR，2015，abs/1502.03167.

[14] HOWARD A G，ZHU M，CHEN B，et al. MobileNets：efficient convolutional neural networks for mobile vision applications [J]. 2017.

[15] DONG X，CHEN S，PAN S J. Learning to prune deep neural networks via layer-wise optimal brain surgeon [J]. Advances in Neural Information Processing Systems，2017，30：4860－4874.

[16] AHN S，HU X S，DAMIANOU C A，et al. Variational information distillation for knowledge transfer [J]. CoRR，2019，abs/1904.05835.

[17] ZOPH，BARRET AND QUOC V. LE. Neural architecture search with reinforcement learning [C] //ArXiv，(2016)：n. pag. abs/1611.01578.

[18] JACOB B，KLIGYS S，CHEN B，et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference [C] //2017.

[19] 杨　春，张睿尧，黄　泷，等. 深度神经网络模型量化方法综述 [J]. 工程科学学报，2023，45 (10)：1613－1629.

[20] LEE S，KIM D，NGUYEN D，et al. Double MAC on a DSP：boosting the performance of convolutional neural networks on FPGAs [J]. IEEE Trans. on CAD of Integrated Circuits and Systems，2019，38(5)：

[21] ZHANG C，LI P，SUN G，et al. Optimizing FPGA-based accelerator design for deep convolutional neural networks [C] //2015.

[22] ZHANG Z，MAHMUD M A P，KOUZANI A Z. Resource-constrained FPGA implementation of YOLOv2 [J]. Neural Computing and Applications，2022，34 (19)：16989－17006.