中北大学学报(自然科学版)
JOURNAL OF NORTH UNIVERSITY OF CHINA(NATURAL SCIENCE EDITION)
2026年 第47卷 第2期(总第226期)
Vol. 47 No. 2 2026(Sum No. 226)

文章编号：1673-3193(2026)02-0234-07 J. North Univ. China, Nat. Sci. Ed., 2026, 47(2): 234-240.

# “错峰”访存优化的低功耗卷积神经网络加速器

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，山西 太原 030051）

**摘 要：**卷积神经网络加速器的推理速度受访存带宽和片上存储资源的限制，在RAM资源较少的FPGA平台上很难充分发挥并行计算的优势。针对这一问题，本文提出了一种采用“错峰”计算和总线动态分配的访存优化策略的加速器结构。首先，采用分层计算和乒乓缓存的方式来减少“高峰”时的访存事务。通过设计灵活的DMA模块实现特征和权重访存总线的动态分配以提高“高峰”时的总线带宽。其次，采用固定乘法阵更改数据流的方式复用乘法阵列来最大化DSP的利用率。然后，采用软硬件协同和高度流水化的设计方式，使加速器从底层到顶层的数据流在时间线上形成全流水结构。模块除锁相环和大型RAM外均采用HDL和源语设计，在XCZU3EG平台资源使用率超过80%的情况下可实现300 MHz的高速运行。实验结果表明：MobilenetV2-1.0-256在所提加速结构上实现了125 GOPs的吞吐率，接近访存总线的带宽上限；推理速度为移动端ARM CPU的7.50倍，为高性能CPU的1.21倍；能效比为桌面级GPU的3.5倍；与其他FPGA上的同类型加速器相比，在性能密度和能效比上均有优势。

**关键词：**FPGA；硬件加速器；卷积神经网络；访存优化；软硬件协同

**中图分类号：**TP183 **文献标识码：**A **doi:** 10.62756/jnuc.issn.1673-3193.2024.04.0014

**引用格式：**画芊昊，李博，杜宸罡．“错峰”访存优化的低功耗卷积神经网络加速器[J]．中北大学学报(自然科学版)，2026，47(2)：234-240.

HUA Qianhao，LI Bo，DU Chengang. Low power convolutional neural network accelerator with memory access optimized by peak shift[J]. Journal of North University of China(Natural Science Edition)，2026，47(2)：234-240.

# Low Power Convolutional Neural Network Accelerator with Memory Access Optimized by Peak Shift

HUA Qianhao，LI Bo*，DU Chengang

(Key Laboratory of Instrumentation Science and Dynamic Measurement (North University of China), Ministry of Education, Taiyuan 030051, China)

**Abstract:** The reasoning speed of convolutional neural network accelerator is limited by memory bandwidth and on-chip storage resources, so it is difficult to give full play to the advantages of parallel computing on FPGA platform with less RAM resources. To solve this problem, this paper proposed an accelerator structure which adopted the memory access optimization strategy of “peak shift” calculation and bus dynamic allocation. Firstly, hierarchical computation and ping-pong caching were used to reduce the number of “peak” access transactions. A flexible DMA module was designed to realize the dynamic allocation of the feature and weight memory access bus to improve the bus bandwidth during “peak” time. Sec-

**收稿日期：**2024-04-15

**作者简介：**画芊昊(1998—)，男，硕士，主要从事深度学习和高性能计算的研究。

***通信作者：**李 博(1972—)，男，副教授，博士，主要从事智能嵌入式系统和数字信号处理的研究。E-mail：libo@nuc.edu.cn。

ondly, the multiplicative array was reused to maximize the DSP utilization by changing the data stream with a fixed multiplicative array. Then, the software and hardware collaboration and highly streamlined design method were adopted to make the data flow from the bottom to the top of the accelerator form a full streamlined structure in the time line. In addition to the phase-locked loop and large RAM, the acceleration module was designed in HDL and source language, and can achieve high speed operation of 300 MHz when the XCZU3EG platform resource utilization exceeds 80%. The experimental results show that MobilenetV2-1.0-256 achieves 125 GOPs throughput on the proposed acceleration structure, which is close to the upper bandwidth limit of the memory access bus. Inference speed of the proposed accelerator is 7.50 times of mobile ARM CPUs and 1.21 times of high-performance CPUs. Compared with desktop GPUs, the energy efficiency ratio is 3.5 times. Compared with the same type of accelerator on other FPGAs, it has advantages in performance density and energy efficiency ratio.

**Key words:** field programmable gate array (FPGA); hardware accelerator; convolutional neural network (CNN); memory access optimization; software and hardware cooperation

## 0 引 言

近年来，卷积神经网络（Convolutional Neural Network，CNN）因其出色的特征提取能力被广泛应用于目标检测、目标识别、视频分类、语音识别和自然语言处理等数据特征量较大的任务之中[1-4]。但是，随着CNN性能的提升，网络的参数规模也变得越来越大，这使得模型的推理对硬件的要求越来越苛刻。然而随着现在物联网的发展，传感器数据量增大，普通嵌入式设备的计算能力和内存带宽无法满足实时推理的需求。因此，研究如何加速卷积神经网络的推理，使其可以部署在功耗更低和体积更小的硬件平台，并实现资源和速度之间的平衡就具有重要意义。

根据硬件平台的不同，硬件层面的加速主要分为中央处理器（Central Processing Unit，CPU）、图形处理器（Graphics Processing Unit，GPU）、专用集成电路（Application-Specific Integrated Circuit，ASIC）和现场可编程门阵列（Field Programmable Gate Array，FPGA）[5]。CPU作为通用数据处理器可以进行各种复杂的运算，但CNN计算的本质是大量重复的乘积累加操作，这使得并行度有限的CPU难以满足移动应用的实时性要求[6]。GPU虽然拥有强大的计算能力，但其较高的功耗和较大的体积不适合在嵌入式平台和边缘计算场景中应用[7]。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优异的计算性能，拥有极高的能效比。但其开发成本高，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[8]。作为一种折中的方案，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。

在资源密度较低的FPGA平台上设计加速器，核心在于提高性能密度和平衡并行度与总线带宽[9]。包振山等[10]提出了一种全流水的加速思路，将整个网络放置到FPGA上进行层间流水处理，网络各层之间同时运算，理论上可以充分发挥出板卡的全部性能。为了降低访外总线的带宽需求和片上存储器的需求，Shang等[11]提出了一种分层调度方案来提高卷积块的性能和灵活性，保证了所有单元的并行运行。Kang等[12]设计了一种基于流的线缓冲区结构，以输入和输出流的流水线方式操作。每个卷积层块都有一个线缓冲区，只存储几行输入数据。线缓冲器的大小与输入图像的宽度成正比，因此，该结构比传统的基于帧的结构所需的中间数据存储量更少。Sang等[13]利用高层次综合（High-level Synthesis，HLS）技术设计了一种基于DSC的高性能CNN加速器，并将其用于低密度FPGA上的图像分类。同时，基于块卷积的自适应数据流调度模式，最小化了中间结果的芯片外数据传输，并提高了数据读取的效率。但是，CNN的层间数据依赖会随着深度的提升逐渐增强，采用全流水结构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变得非常复杂，各层的资源分配难以平衡。Shang和Kang的团队所使用的VC709和XCKU5P与其所部署的Mobilenet网络结构相比本身就是资源过剩的。Sang采用的高级综合技术对FPGA内部资源的利用很难达到寄存器传输级（Register Transfer Level，RTL）

的灵活性，适合做一些快速验证的工作。

针对以上问题，本文提出了一种采用“错峰”访存优化策略的低功耗卷积神经网络软硬件协同加速结构。主要的工作如下：1）通过分析网络的特点，提出一种基于“错峰”计算和总线动态分配的访存优化方案。对于网络浅层和深层，采用不同的计算策略来尽可能减少“高峰”时的访存事务。同时，设计灵活的直接存储器访问(Direct Memory Access，DMA)模块实现feature_ram和weight_ram访存总线的动态分配，以提高“高峰”时的总线带宽。2）重封装双乘法器实现双int9乘法，且采用固定乘法阵更改数据流的方式复用乘法阵列来最大化数字信号处理(Digital Signal Processing，DSP)器件的利用率。3）采用软硬件协同的形式，在ARM端部署ARMNN推理框架来处理输入层计算，使得ARM与FPGA在时间线上也形成一种流水结构。同时，开发桌面端C++应用程序，自动根据不同的输入尺寸生成加速器顶端中枢模块所需的参数。

## 1 加速器设计

### 1.1 加速器整体访存结构和数据流设计

访存一直是制约FPGA中加速器设计的一个瓶颈。FPGA的片上高速存储资源有限，很难完全缓存模型推理过程中产生的中间结果或网络完整的权重信息。硬件的瓶颈是客观存在的，访存策略的设计思想是尽可能减少访存事务的次数，并且尽可能释放总线带宽，以提高加速器可设计的并行度的上限。

#### 1.1.1 数据搬运模块DMA

Zynq MPSOC中PS的Slave端有4条AXI_HP总线，采用突发读写的形式，数据位宽设置最大为128 bit。DMA由AXI协议层的实现模块、ram和控制逻辑组成，内部结构如图1所示。核心的控制逻辑连接协议层和ram块，以DMA_feature1为例，控制逻辑收到读取命令(命令格式：读取的地址(ADDRR_DDR)，突发次数，突发长度，写入地址(ADDRW_ram)和是否使能ram写端口的非DMA操作)时，每从总线接收到数个数据就拼接成一个256位的数据，然后从ADDRW_ram顺序写入feature_ram11中。当控制逻辑接收到写命令(命令格式：读取的地址(ADDRR_DDR)，突发次数，突发长度，写入地址(ADDRW_ram)和是否使能ram读端口的非DMA操作)时，控制逻辑从feature_ram22的ADDRR_ram开始顺序读取数据，并将数据拆分成数个128 bit数据，然后通过AXI写通道从ADDRW_DDR顺序写入。ram的读写端口非DMA操作使能可以独立决定DMA中的ram块是被AXI总线占用还是可以被外部模块当作普通ram来访问，其他的DMA_weight和DMA_res功能与上类似。

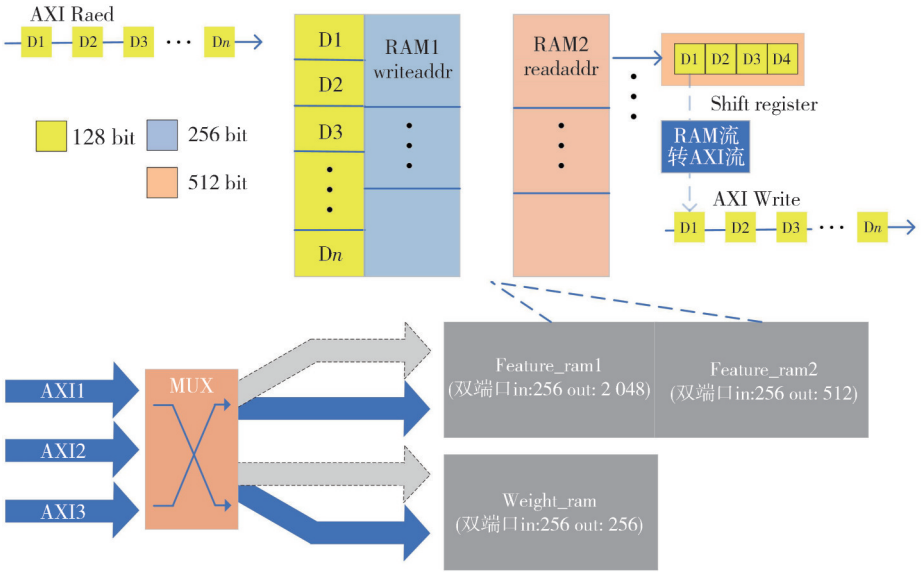


图1 DMA内部结构

Fig. 1 Internal structure of DMA

feature和weight的输入缓存采用真双端口设计，DMA的控制模块可以动态决定feature_ram和weight_ram占用的AXI总线个数，最多支持两条总线同时对ram的两个不同地址进行写操作(采用乒乓缓存的操作，意味着输入缓存的ram不会被同时进行读和写的操作)。feature输出缓存的读端

口采用双端 512 bit，且整个系统只有中间 feature 需要占用 AXI 的写通道，DMA 模块可以动态设置 feature 的输出缓存占用的总线数（理论上 4 条总线的写通道都可以由控制输出缓存的 DMA 控制）。

网络推理过程中，feature 和 weight 访存的“高峰”会出现在不同的阶段，本文设计的 DMA 结构可动态分配总线资源，尽可能释放访存“高峰”期间的总线带宽，且寻址方式自由，内部 ram 可以被外部模块当作普通 ram 直接访问，可以大量节省 BRAM 资源。

### 1.1.2　“错峰”的访存数据流优化

本文采用一种“错峰”的访存数据流优化方法来尽可能减少“高峰”时的访存事务。Zynq MPSOC 中每条 AXI_HP 总线的最大传输带宽为 128/8×300×$10^6$（时钟）=4 800 MB/s，正好对应 PS 端每块 DDR4 的最大数据速率（美光 MT40A512M16LY-062E×4）。本文通过搭建测试平台对三条总线进行读写测试（突发长度为 256 且突发间的地址并不连续）。由图 2 可以看出，两次突发之间会有 30 个左右的时钟间隔（从发送读写地址到收到回应），且一次突发内的“毛刺”现象会随着总线使用数量的上升而上升。多次实验表明总线的实际性能只能达到理论值的一半左右，这也是后续 DPU 并行度设计的依据。

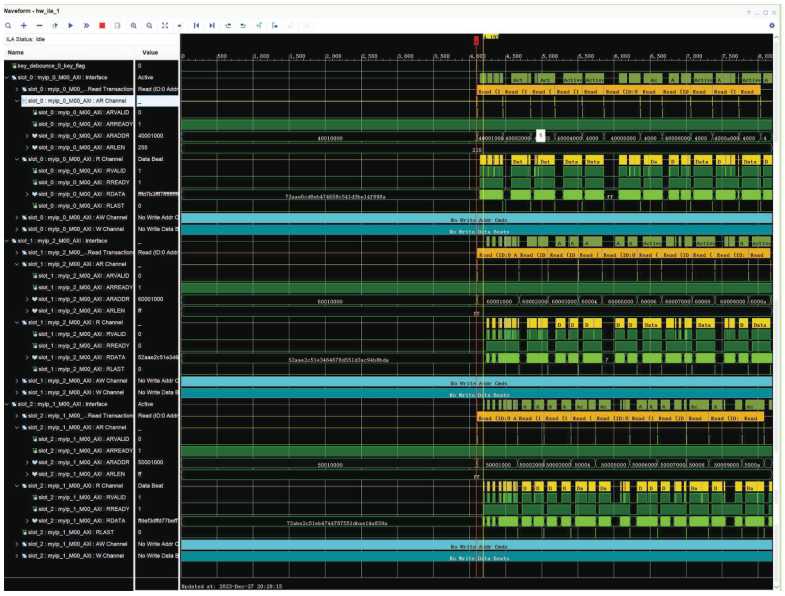

图 2　AXI 三通道同时读写的结果

Fig. 2　Simultaneous reading and writing results at three channels of AXI

本文通过分析网络推理的特点发现网络在浅层（PointWise，PW）计算期间的访存压力主要来自于 feature（深层（DepthWise，DW）由于其自身计算的特点，feature 和 weight 只需要各读取一遍便可输出所有结果），而在网络层数加深和采样倍数增加的情况下，访存的压力会由 feature 转为 weight（浅层时 weight 较小，可以整个存在片上 ram 中供 DPU 来回读写。深层时 feature 较小，可以整个存在片上 ram 中供 DPU 来回读写）。根据这一特点，本文加速结构在浅层 PW 时采用固定 feature 循环 weight 的形式进行计算，在深层时采用固定 weight 循环 feature 的方式进行推理，以这种“错峰”的方式减少“高峰”时的访存事务，优化访存数据流，保证输入尺寸在 512 以下时，“高峰”时每层计算只从片外存储器中读取一遍数据。

## 1.2　核心计算单元 DPU 的设计

本文采用的是一种并行度为 128 的 DSP 固定乘法阵列，通过改变输入数据流和输出数据流的方式来实现多种卷积操作。本文通过重新封装双乘法器[14]使单个 DSP48e2 可以实现双 int9 乘法，以同时支持对称和非对称量化，其原理如图 3 所示。

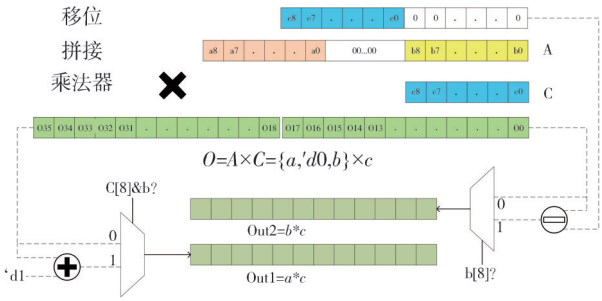


图 3　双 int9 封装

Fig. 3　Dual int9 package

计算开始时，feature_buffer1 和 weight_buffer1 从 feature_ram 和 weight_ram 中读取若干数据（DW 卷积时 feature_buffer 为滑动窗结构），同时 fea-

ture_buffer2和weight_buffer2读取下轮计算所需数据；然后，feature_ buffer与weight_buffer中的数据同时送入乘法阵列；接着，乘积的结果与偏置数据进入后处理模块(DW与PW的加法阵)；最后，后处理模块得到的结果经过量化和四舍五入后送入save_buffer，由DMA模块按照DW和PW两种模式将结果写入外部缓存。上述数据流采用流水线设计同时进行，所提到的所有模块也采用块内流水形成全流水形式，使DPU的计算效率实现最大化。

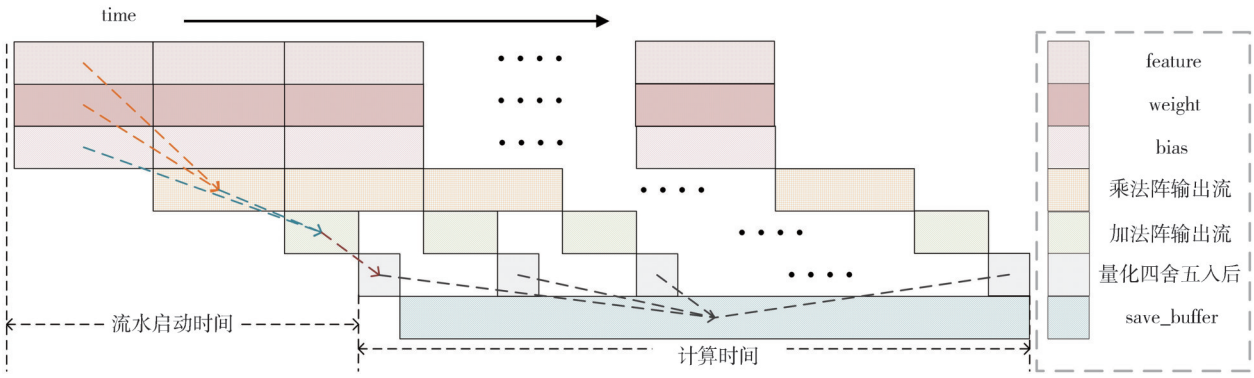


图4 DPU内数据流

Fig. 4 Data flow in DPU

### 1.3 软硬件协同设计

本文采用软硬件协同设计的形式，在ARM端部署ARMNN-C++推理框架处理输入层计算，使ARM与FPGA在时间线上也形成一种流水结构。同时，开发桌面端C++应用程序，自动根据不同的输入尺寸生成加速器顶端中枢模块所需的参数。软件部分更多承担的是辅助性的计算任务、任务调度、推理验证和数据预处理等。

## 2 实验设计

在Windows环境下进行加速器的整体开发，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(部署平台为XCZU3EG FPGA+ARM Cortex-A53，ILA需消耗大量BRAM，故调试平台采用XCZU15EG FPGA+ARM Cortex-A53)。

### 2.1 横向对比实验

通过对比加速器与嵌入式ARM CPU、AMD移动端CPU和NVIDIA移动端GPU的推理速度和能效比来验证本文设计的实用价值。其中，嵌入式ARM CPU直接使用的是XCZU3EG的PS部分；桌面级CPU使用的是AMD的Ryzen 7 6800H with Radeon Graphics，使用pycharm在Keras-GPU环境下(os. environ[“CUDA_VISIBLE _DEVICES”]=“-1”)计算5 211幅图输入的运算结果；移动端GPU使用的是NVIDIA GeForce RTX 3060 Laptop GPU，使用pycharm在Keras-GPU环境下计算5 211幅图输入的运算结果。

### 2.2 纵向对比实验

通过与国内同样基于FPGA的卷积神经网络加速器的设计方案进行对比来验证本文所设计加速器的优势。由于不同硬件平台间的资源和功耗存在较大差异，因此通过增加性能密度(GOPs/LUT和GOPs/DSP)两个指标来充分验证加速器的性能。

## 3 性能评估

### 3.1 横向对比

ARM端的测试环境为Linux+ARMNN-C++，程序运行时ARM核将一个batch的图片文件和模型的权重文件从SD卡中读入DDR。在ARMNN框架下加速推理100幅输入图片，测得ARM端的平均运行时间为48 ms，功耗为2.6 W。

CPU和GPU使用在Keras下修改MoblienetV2输入层为256×256后重训练得到的模型，是未经量化的版本(float32而非uint8)。由于Tensorflow下量化之后的模型转为tflite，tflite与桌面端GPU兼容性差，因此其运行结果较差，其次即使在8 bit量化推理下，其中间过程的运算

依旧是采用float32型，量化只是减小了原始权重和输入图像在内存或显存中的占用空间，量化操作对桌面端CPU和GPU的推理效果不理想，故CPU和GPU使用的模型为Keras下的模型格式（除数据存储格式外，网络的结构与ARM和FPGA所使用的模型一致）。其次，本实验关注的重点是加速模型推理，重训练后的MoblienetV2-float参数不一定为最优解，正确率不影响本次实验对推理速度的验证，故不再给出模型训练阶段的详细参数。

CPU平均单帧时间为7.75 ms，GPU在同样输入下得到的单帧时间为1.08 ms。如图5所示，在CPU执行推理计算任务期间，设备功耗维持在45 W，GPU的功耗也稳定在95 W左右。

本文加速器的运行环境与ARM端测试时相同，区别在于增加了FPGA部分的设计。如表1所示，本文加速器的推理速度为嵌入式ARM CPU的7.50倍，为高性能CPU速度的1.21倍且功耗仅为其1/10。相比桌面级GPU，本文加速器在吞吐量上略微落后，但能效比却达到了桌面级GPU的3.5倍。上述结果证明本文的硬件加速器具有高性能密度和高能效比的优点，同时4.5 W的低功耗（其中2.7 W来自于ARM核）和125 GOPs的吞吐量，使其完全满足边缘计算场景的要求。

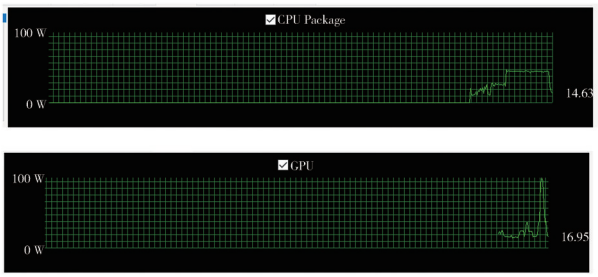


图5　功耗曲线

Fig. 5　Power consumption curve

表1　加速器横向对比

Tab. 1　Lateral contrast of accelerator

| 类别 | 推理延时/ms | 功耗/W | 时钟/GHz | 吞吐量 | 能效比 |
|---|---|---|---|---|---|
| 嵌入式ARM | 48.00 | 2.70 | 1.2 | 16.66 GOPs | 6.17 GOPs/W |
| CPU | 7.75 | 45.00 | 3.2 | 103 FLOPs | 2.28 FLOPs/W |
| GPU | 1.08 | 95.00 | 1.4 | 740 FLOPs | 7.78 FLOPs/W |
| 本文加速器 | 6.40 | 2.70+1.87 | 0.3 | 125 GOPs | 27.35 GOPs/W |

## 3.2　纵向对比

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）、性能密度（GOPs/LUT、GOPs/DSP）和片上RAM资源四个指标来综合验证加速器的性能。

由表2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。由于本文采用了复用乘法阵列和重封装双乘法器的设计策略，其GOPs/DSP在对照组中最高。文献[10]的GOPs/LUT略高于本文，但其采用的是4bit量化策略且输入尺寸小于本文加速器。文献[17]的能效比较高是由于ZYNQ7000系列PS部分的动态功耗比Ultra+系列低1.2 W左右，但其资源消耗量远高于本文所设计加速结构，且其硬件平台的资源体量对于所部署的网络结构来说是资源过剩的。

表2　加速器纵向对比

Tab. 2　Longitudinal contrast of accelerator

| 参数项 | 参数组 | | | | | |
|---|---|---|---|---|---|---|
| | 文献[10] | 文献[15] | 文献[13] | 文献[16] | 文献[17] | 本文 |
| FPGA | XCZU3EG | XCZU3EG | XC7Z020 | XCZU7EV | XC7Z045 | XCZU3EG |
| Model | Ultranet | Mobilenet | MobilenetV2 | YOLOV2 | VGG−16 | MobilenetV2 |
| Frame size | 320×160 | 128×128 | 224×224 | 416×416 | 224×224 | 256×256 |
| Precision/bit | $4\times10^3$ | $8\times10^3$ | $8\times10^3$ | $8\times10^3$ | $16\times10^3$ | $8\times10^3$ |
| LUTs | $4.2\times10^4$ | $5.8\times10^4$ | $4.1\times10^4$ | $9.1\times10^4$ | $1.25\times10^5$ | $5.2\times10^4$ |
| DSP | 360 | 224 | 206 | 387 | 449 | 194 |
| Block RAM/KB | 660 | 972 | 576 | 1037 | 2385 | 576 |
| Throughput/(GOP•s$^{-1}$) | 126 | 96 | 10.8 | 100 | 115 | 125 |
| Power/W | 6.65 | 4.15 | 3.1 | 4.8 | 3.8 | 4.57 |
| PowerEfficiency/(GOPs•W$^{-1}$) | 18.9 | 23.4 | 3.5 | 20.8 | 30.2 | 27.3 |
| GOPs/LUT | $3.0\times10^{-3}$ | $1.6\times10^{-3}$ | $2.6\times10^{-4}$ | $1.1\times10^{-3}$ | $9.2\times10^{-3}$ | $2.4\times10^{-3}$ |
| GOPs/DSP | 0.35 | 0.39 | 0.05 | 0.25 | 0.26 | 0.64 |

# 4　结　论

本文设计了一种边缘计算场景下采用“错峰”访存优化策略的低功耗卷积神经网络软硬件协同加速结构。较低的DSP和LUT消耗量使其可以部署在大多数的FPGA中，整个加速器的功耗只有4.57 W。

但值得注意的是，由于采用了软硬件协同的设计方法，PS部分的功耗是计算在总功耗内的。在大多数系统中，CPU或单片机是不可或缺的，需要承担任务调度与运行嵌入式应用系统的任务，本文加速器并未全部占用PS部分的四个大核，所以理论上来说，在实际应用中富余的PS部分完全可以承担更多其他任务，这也是基于ZYNQ平台的加速方案相比纯FPGA平台的优势，理论上加速器在实际应用中的功耗应小于标注功耗。本文加速器的推理速度为嵌入式ARM CPU的7.50倍，能效比达到了桌面级GPU的3.5倍。与其他基于FPGA的同样先进的加速器设计相比，本文所设计的加速器在能效比和性能密度方面均有优势，具有较高的应用价值。

## 利益冲突声明/Conflict of Interests

所有作者声明不存在利益冲突。

All authors declare no relevant conflict of interests.

## 参考文献：


[1] REDMON J. FARHADI A. YOLOV3：An incremental improvement[DB/OL].(2018-04-08)[2024-04-14]. https://arxiv.org/abs/1804.02767.

[2] SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition[DB/OL].(2015-04-10)[2024-04-14]. https://arxiv.org/abs/1409.1556.

[3] KARPATHY A, TODERICI G, SHETTY S, et al. Large-scale video classification with convolutional neural networks[C]//2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014：1725-1732.

[4] ABDEL-HAMID O, MOHAMED A R, JIANG H, et al. Convolutional neural networks for speech recognition[J]. IEEE/ACM Transactions on Audio, Speech, Language Processing, 2014, 22(10)：1533-1545.

[5] CAPRA M, BUSSOLINO B, MARCHISIO A, et al. Hardware and software optimizations for accelerating deep neural networks：Survey of current trends, challenges, and the road ahead[J]. IEEE Access, 2020, 8：225134-225180.

[6] GUO K, ZENG S, YU J, et al. A survey of FPGA based neural network accelerator[DB/OL].(2018-12-06)[2024-04-14]. https://arxiv.org/abs/1712.08934.

[7] MENG C, DAI H. A hardware-independent time estimation method for inference process of convolutional layers on GPU[J]. Performance Evaluation, 2023, 162：102368.

[8] KELLER B, VENKATESAN R, DAI S, et al. A 95.6-TOPS/W deep learning inference accelerator with per-vector scaled 4-bit quantization in 5 nm[J]. IEEE Journal of Solid-State Circuits, 2023, 58(4)：1129-1141.

[9] WILLIAMS S, WATERMAN A, PATTERSON D. Roofline：An insightful visual performance model for multicore architectures[J]. Communications of the ACM, 2009, 52(4)：65-76.

[10] 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.
BAO Zhenshan, GUO Junnan, ZHANG Wenbo, et al. UltraAcc：A customized low power and high performance CNN accelerator with dataflow on FPGAs[J]. Chinese Journal of Computers, 2023, 46(6)：1139-1155.(in Chinese)

[11] SHANG J W, ZHANG K, ZHANG Z, et al. A high-performance convolution block oriented accelerator for MBConv-Based CNNs[J]. Integration, 2023, 88：298-312.

[12] KANG H J, YANG B D. AoCStream：All-on-chip CNN accelerator with stream-based line-buffer architecture and accelerator-aware pruning[J]. Sensors, 2023, 23(19)：8104.

[13] SANG X T, RUAN T, LI C L, et al. A real-time and high-performance MobileNet accelerator based on adaptive dataflow scheduling for image classification[J]. Journal of Real-Time Image Processing, 2023, 21(1)：4.

[14] LEE S, KIM D, NGUYEN D, et al. Double MAC on a DSP：Boosting the performance of convolutional neural networks on FPGAs[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2019, 38(5)：888-897.

[15] PHAM-QUOC C, THINH T N. Efficient FPGA-based convolutional neural network implementation for edge computing[J]. Journal of Advances in Information Technology, 2023, 14(3)：479-487.

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

[17] ZHANG C, WANG X A, YONG S S, et al. An energy-efficient convolutional neural network processor architecture based on a systolic array[J]. Applied Sciences, 2022, 12(24)：12633.