DOI: 10.19650/j.cnki.cjsi.J2107514

嵌入式神经网络加速器及 SoC 芯片

易冬柏[1],陈 恒[2],何乐年[1]
(1. 浙江大学信息与电子工程学院 杭州 310007; 2. 珠海零边界集成电路有限公司 珠海 519000)

摘 要:为了提高人工智能加速器的运算效率和功耗效率,提出了一种新的卷积神经网络(CNN)加速器结构,并实现了神经网络存算一体的方法。首先,设计出一种神经网络架构,其具有高度并行计算以及乘加器(MAC)单元高效运行的特性。其次,为了降低功耗和面积,采用了对称的静态随机存储器(SRAM)阵列和可调数据流向结构,实现多层网络在 SRAM 中高效计算,减少了访问外部存储器次数,降低了功耗,提高运算效率。通过中芯国际 40 nm 工艺,完成了系统芯片(SoC)设计、流片与测试。结果表明运算速度在 500 MHz 下,算力可达 288 GOPS;全速运行功耗 89.4 mW;面积 1.514 mm^2;算力功耗比 3.22 TOPS/W;40 nm 算力面积比为 95.1 GOPS/mm^2。与已有文献的相比,算力功耗至少提升 4.54%,算力面积至少提升 134%,对于嵌入式场景应用较适合。

关键词: 人工智能;加速器;卷积神经网络;边缘侧;卷积神经处理器
中图分类号: TH166 TN47 TP391.4 **文献标识码**: A **国家标准学科分类代码**: 510.4 510.3

Embedded neural network accelerator and SoC chip

Yi Dongbai[1], Chen Heng[2], He Lenian[1]
(*1. College of Information Science & Electronic Engineering, Zhejiang University, Hangzhou 310007, China;*
2. Zhuhai Edgeless Semiconductor Co., Ltd., Zhuhai 519000, China)

Abstract: In order to improve the operation efficiency and power efficiency of artificial intelligence accelerator, proposes a new convolutional neural network (CNN) accelerator, and realizes a computing-in-memory method. Firstly, a neural network architecture is designed, which has the characteristics of highly parallel computing and efficient operation of MAC unit. Secondly, in order to reduce power consumption and die size, a symmetric SRAM array and an adjustable data flow structure are adopted to realize the efficient computation of multi-layer network in SRAM, which reduces the times of external memory access and the power consumption of SoC system. Operation efficiency is improved as well. Through the 40 nm process of SMIC, the SOC design, tape and test are completed. Results show that the computational power can reach 288 GOPS at 500 MHz, the power consumption at full speed is 89.4 MW, the area is 1.514 mm^2, the computational power consumption ratio is 3.22TOPS/W and the 40nm computational power area ratio is 95.1 GOPS/mm^2. Compared with results in other literatures, the power consumption and area of computing power increase by at least 4.54% and 134%, respectively, which is more suitable for embedded ends.

Keywords: artificial Intelligence; accelerator; convolutional neural networks; edge; convolutional neural processor

0 引 言

人工智能正在改变我们的世界,不仅改变人类的生活,也正在改变人与人的互动方式。但在与人更深入接触的生活中,人工智能能否拥有更广泛的使用,就需要更贴近生活的产品诞生。人工智能算法与硬件相结合的嵌入式系统可应用于不同的应用场景。如何让人工智能,特别是像物体检测,人形识别,人脸识别技术更好的符合实际的应用[1-4],就需要更具性价比的算法与芯片。

收稿日期:2021-02-22 Received Date: 2021-02-22

当前有不同的方法实现人工智能,如边缘端进行图像获取,云端进行推理判断,再下发给边缘端进行最终的判断,或者两者进行一定的结合。在处理器方面,如中央处理器(CPU)/图形处理器(GPU)运算完成典型的卷积神经网络[5],也有如 FPGA 完成运算量较小的神经网络[6-7]。但是,对于嵌入式方向来说 CPU/GPU 或者 FPGA 来实现,实用价值较弱。使用专用芯片设计(ASIC)方式,或者 SoC 方式是当前最佳的选择。

当前的深度学习 SoC 芯片大多侧重于最高运算性能,且大部分运用在云端的运算。用在边缘侧的人工智能芯片侧重于运算性能,功耗较高,成本也不低[8-10]。在现实生活中,有着大量的人工智能运用的场景,但不需要强大的算力,仅需要适当的算力加上足够低的功耗和成本,这样的场景非常适合嵌入式人工智能芯片。嵌入式芯片要求功耗,性能,面积的平衡性,达到较高的算力功耗比,和算力面积比。当前嵌入式领域较多的应用仍然是 FPGA 或者应用处理器进行处理计算,FPGA 性能不能满足要求,应用处理器又有着太高的功耗和成本。嵌入式领域,需要更高性能,更小的功耗和面积。如何使用最优化的结构和数据交换是嵌入式人工智能 SoC 设计的一大挑战。

针对上述问题,本文提出了轻量化的基于 8 bit、16 bit 定点计算精度的神经网络加速器。以并行的运算单元搭配高性能的数据流控制,实现高效率的卷积神经网络存储计算一体化设计。同时加上嵌入式的 SoC 设计,得以在智能的物联网应用中发挥功效。当工作频率在 500 MHz 下,计算精度为 16 bit 定点时,卷积神经网络加速器的运算性能达到 288 GOPS,实现了较高的算力功耗比和算力面积比。

1 卷积神经网络

卷积神经网络是一种具有共享权重结构、平移不变性特征的人工神经网络,其主要由卷积层、池化层和全连接层组成。卷积层的作用是通过权值共享来提取特征,池化层则主要是为了降低数据维度与防止图像过拟合,全连接层的作用是把最后一层卷积的输出图像由高维变成低维,并且把输入的信息进行提取整合,再经过激活函数的映射,以实现特征到标签的映射。图 1 为一种典型的卷积神经网络[8]。

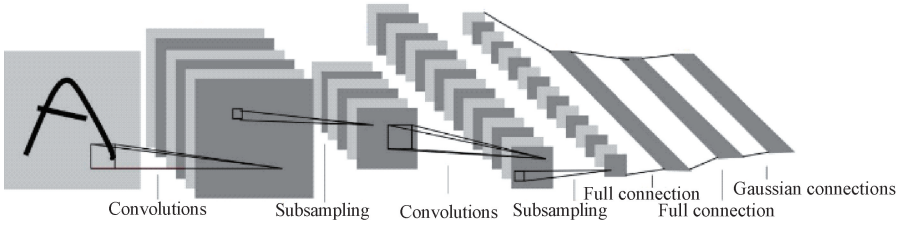


图 1 典型 Lenet 卷积神经网络

Fig. 1 Lenet convolutional neural network

一个卷积神经网络是为二维形状而特殊设计的多层感知机,这种二维形状对平移,比例缩放,倾斜或者其他形式的变形具有高度不变性。网络的结构包括如下一些约束[5]:

1)特征提取。一旦一个特征被提取出来,只要相对与其他特征的位置被近似的保留下来,它的精确位置就变得没有那么重要了

2)特征映射。网络的每个计算曾是多个特征映射组成的,每个特征映射都是平面形式的。

3)子抽样。每个卷积层跟着一个实现局部平均和子抽样的计算层,由此特征映射的分辨率降低。

CNN 的主要运算是卷积转换输入特征映射的操作(I)使用权重(W)输入输出特征映射(O)。基本卷积运算总是可以写成如下算式:

$$O[f][x][y] = \sum_{c=0}^{C}\sum_{i=0}^{K}\sum_{j=0}^{K} I[c][x+i][y+i] \times W[f][c][i][j] \quad (1)$$

式中:C 是输入通道的数量;K 是内核的大小;O 是输出图像上的输出特征;W 是权重值;I 是输入图像的特征值。

2 硬件实现及其结构

在边缘测的人工智能芯片设计中,受限于芯片成本、功耗因素,神经网络处理器不能堆积过多的 MAC 单元来实现高算力,而要在有限 MAC 单元情况下,提高 MAC 的计算效率和利用率,实现算力、面积及功耗的折中。另外,在卷积神经网络的计算过程中,会产生大量的临时数据(Psum)及隐含层数据,如果把这些数据都放在外部 DDR 存储器的话,不但会极大的降低神经网络计算效率,还会显著增加系统功耗及 SOC 系统的带宽,所以在电路设计上通常会采用 SRAM 作为数据缓存来提高计算效率和减少 DDR 的读写次数,同样受限于芯片成本因素,SRAM 容量不能设计的过大,如何对 SRAM 进行结构划分及数据处理直接影响到神经网络处理器的计算效率

及功耗水平。

本文针对边缘测人工智能芯片的特点,提出了一种神经网络处理器架构,实现神经网络的高度并行计算及 MAC 单元高效率运行,并且设计对称结构 SRAM 阵列和可调数据流向实现多层网络在 SRAM 中高效计算,极大减少了访问外部 DDR 存储器次数,有效降低了 SOC 系统功耗,设计专用于神经网络的特征图读取和写出电路,支持任意大小特征图的自动裁剪和拼接操作,实现在 SRAM 不能缓存所有特征图情况下的 SRAM 和 DDR 的高效数据搬移,达到存储计算一体化。本文所提出的卷积神经网络处理器(convolution neural network processor, CNP)架构如图 2 所示。

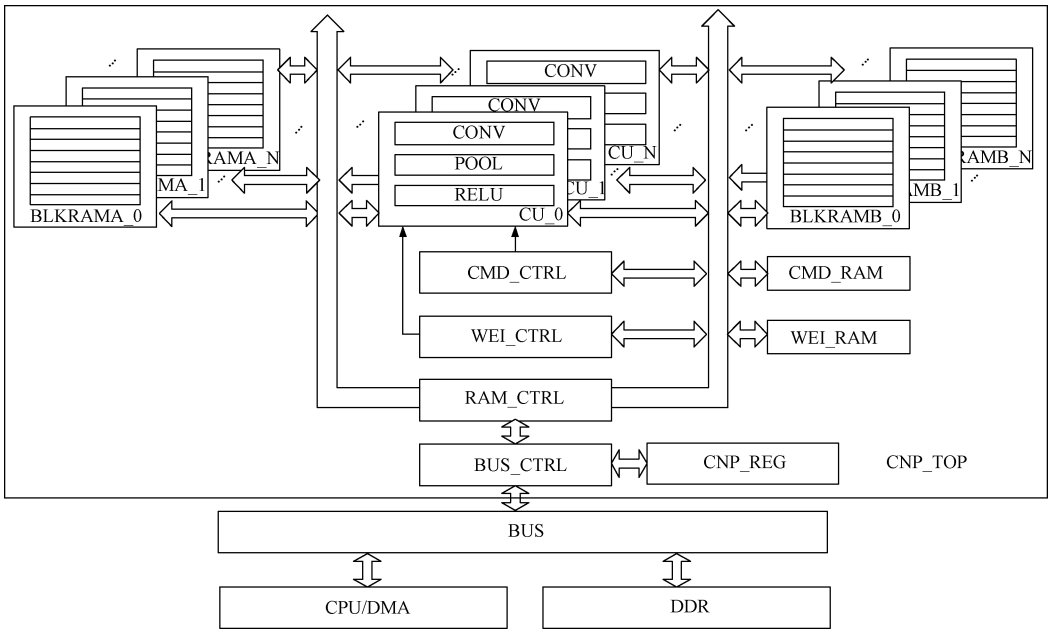


图 2　卷积神经网络处理器架构图

Fig. 2　Convolution neural network processor architecture

2.1　卷积神经网络处理器硬件架构及设计

主要由以下几个模块构成:

1)BLKRAM SRAM 阵列:缓存输入层/输出层/隐含层的特征图,分为两个部分,BLKRAMA 和 BLKRAMB,BLKRAMA 或 BLKRAMB 又可分为 *N* 个子 BLKRAM,如图所示,每个子 BLKRAM 可分别存储不同的特征图;

2)卷积神经运算单元(computing unit, CU)阵列:卷积神经网络计算模块,由 N 个卷积神经网络单元(CU)构成,每个 CU 单元由 CONV/POOL/RELU 3 个子模块构成,分别进行卷积/池化/激活运算,并且每个 CONV/POOL/RELU 子模块都包含 8 路并行计算单元,用于实现神经网络的高算力及高效率;

3)指令控制模块(CMD_CTRL)和指令存储模块(CMD_RAM):指令控制模块从 CMD_RAM 中读取卷积神经网络的操作指令(如卷积指令,池化指令等),产生控制信号,用以控制各模块按照指令运行;

4)权值控制模块及权值缓存 RAM(WEI_RAM):权值控制模块读取 WEI_RAM 中缓存的权值数据,并将其传送给 CU 模块进行卷积计算;

5)RAM 控制模块:控制所有 RAM(BLKRAM 阵列/CMD_RAM/WEI_RAM)的读写接口;

6)总线控制模块(BUS_CTRL):用以控制系统总线的读写,此模块包含 slave 总线和 master 总线,slave 总线用于 CPU 对所有 RAM 读写操作及寄存器控制,master 总线用于硬件自动读取指令、模型以及特征图读写;

7)CNP 寄存器模块:通过 BUS_CTRL 模块的 slave 总线接收 CPU 配置信息,如神经网络指令源地址 cmd_src_addr,指令长度(cmd_length),中断功能及 CNP 使能功能,以及反馈 CNP 的状态信息。

2.2　卷积神经网络流程

本文所提出的 CNP 架构可以在 CPU 极少参与的情况下,高效率的完成卷积神经网络计算,完成存储计算一体化,在极大提升计算效率同时完全释放出 CPU 资源,工作流程如下:

1)CPU 配置 CNP 寄存器,配置卷积神经网络指令的首地址, 指令长度,打开 CNP 中断功能,使能 CNP;

2)BUS_CTRL 根据指令长度和指令首地址配置通过总线从 DDR 上面读取神经网络指令数据,并传送给 RAM_CTRL 模块,RAM_CTRL 接收到指令数据将其存储到 CMD_RAM 中,并使能 CMD_CTRL;

3)CMD_CTRL 模块读取 CMD_RAM 中的指令,将其解析出来并执行,本文的神经网络指令包括初始化指令、特征图读取指令、卷积运算指令、池化运算指令、激活指令、循环指令、特征图写出指令,结束指令等,以初始化指令->特征图读取指令->卷积运算指令->池化运算指令->激活指令->特征图写出指令->结束指令序列为例,CNP 的运行过程如下:

1)解析初始化指令,将输入/输出特征图的尺寸及数量、模型权值地址、计算精度等初始化信息解析出来;

2)特征图读取指令,触发 BUS_CTRL 从 DDR 上读取输入特征图传送给 RAM_CTRL,RAM_CTRL 根据特征图信息将特征图摆放到 BLKRAM 阵列(如 BLKRAMA_0-BLKRAMA_N)的对应位置;

3)执行卷积运算指令,在此指令工作模式下,RAM_CTRL 模块将特征图按照顺序从 BLKRAMA 阵列读取出来传送给 CU 阵列,于此同时 BUS_CTRL 从 DDR 中读取模型权值存储至 WEI_RAM,WEI_CTRL 模块读取 WEI_RAM 权值也传送给 CU 阵列,CU 阵列在接收到特征图及权值数据后启动 CONV 模块进行大量的并行卷积计算,将结果通过 RAM_CTRL 存储至 BLKRAMB 阵列;

4)执行池化运算指令,在此指令工作模式下,RAM_CTRL 模块将特征图按照顺序从 BLKRAMB 阵列中读取出来送给 CU 阵列,CU 阵列启动 POOL 模块进行并行池化计算,将结果通过 RAM_CTRL 存回 BLKRAMB 阵列;

5)执行激活运算指令,类似池化指令,RAM_CTRL 模块读取 BLKRAMB 阵列的特征图送给 CU 阵列,CU 阵列启动 RELU 模块进行激活运算并把结果存储至 BLKRAMB 阵列中;

6)执行特征图写出指令,RAM_CTRL 模块将计算后的特征图从 BLKRAMB 中读取出来并传送给 BUS_CTRL 模块,BUS_CTRL 将其写出至 DDR 的对应地址空间上;

7)执行结束指令,停止所有模块,发出结束状态,发起中断。

2.3 神经网络运算结构及数据流

CNP 在卷积神经网络运算时以网络层为单位进行计算,如果一个卷积网络有 4 层,CNP 会先计算网络的第 1 层,然后再计算第 2 层,依次逐层计算直到最后一层。以网络的某一层卷积计算为例,假设输入特征图(IN_CH)有 N 个,输出特征图(OUT_CH)有 M 个,首先 CNP 会按顺序从 BLKRAM 中读取 N 个 IN_CH,将其与 N 个权值(CH W)进行卷积,再把卷积结果累积加起来得出一个 OUT_CH,将其存储在另外的 BLKRAM 上,以上运算循环 M 次即可得到 M 个 OUT_CH,如图 3 所示。

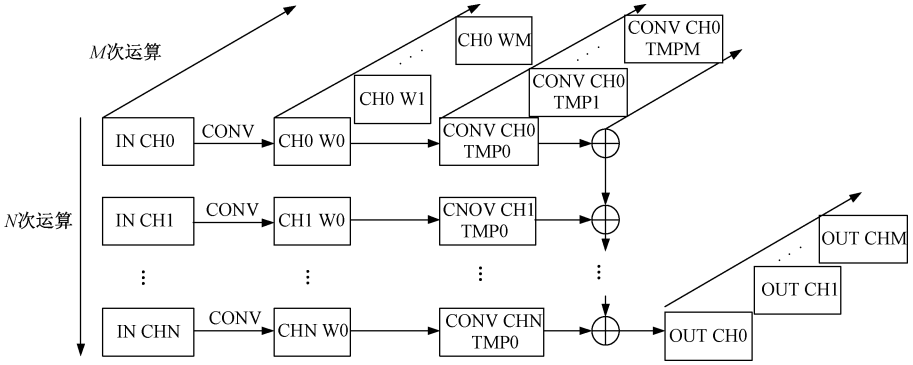


图 3 单层卷积运算结构

Fig. 3 Single layer convolution computing architecture

当 CU 阵列和 BLKRAM 阵列为 4 时,如图 4 所示,单次卷积运算流程为同时从 4 个 BLKRAMA 中同时读取 4 个 IN_CH 的数据,将数据传送给 4 个 CU 单元的 CONV 运算模块,4 个 CONV 模块同时对这 4 个 IN_CH 进行卷积运算,输出 4 个临时卷积图像(CONV_TMP0,CONV_TMP1,CONV_TMP2,CONV_TMP3)后将其数据相加变成一个临时卷积图像(CONV_TMP),读取 BLKRAMB 上一次卷积操作时存储的临时图像(CONV_PSUM)与 CONV_TMP 累加后再存回 BLKRAMB 的同一地址作为下一次的 CONV_PSUM。以此种方式循环 $N/4$ 次后在 BLKRAMB 中可以得到一个 OUT_CH。再将上述运算再次循环 M 次后可以得到本层网络所有的 OUT_CH。

由于卷积神经网络中超过 90% 的计算量都集中在卷积层中,为了提高计算速度,每个 CU 单元的 CONV 模块使用 8 路并行卷积计算的方式,4 个 CU 同时工作,实现总计 32 路并行卷积计算,以 3×3 卷积为例,每路卷积设计 9 个 MAC 单元,总共实现 288 个 MAC 单元同时并行计算。为了进一步实现 MAC 单元的高效率计算,把每个 BLKRAM 划分为为 8 块子 BLKRAM,如图 2 中的 BLKRAMA_0 所示,每个子 BLKRAM 分别存储一行图像,在此结构下,每个 CONV 模块可以同时无间断的读取 8 行图像,实现 MAC 单元近乎满效率计算,最终达到每个

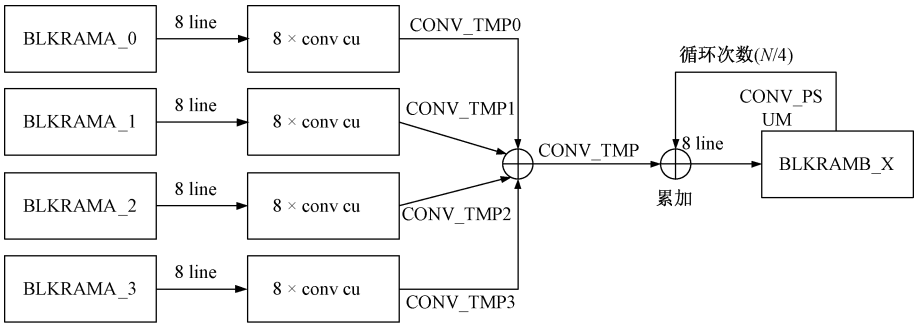


图 4　单次卷积运算结构及数据流向

Fig. 4　Convolution computing architecture and data flow

时钟周期完成 288 个 MAC 计算的速率。

本文所提出的卷积运算结构不要单独的 PSUM 存储单元,而是将输出缓存直接作为 PSUM 存储区,即节省了电路面积,又提高了数据利用率。

当 CNP 计算网络的激活层时,由于输入和输出特征图数量相同,并且输入特征图只会使用一次,所以在进行激活层运算时,在 BLKRAM 读出输入特征图后,进行激活计算后把结果写回到原输入特征图的存储地址。如图 5 所示,假设激活层有 M 个输入特征图和输出特征图,在计算激活层时,同时从 BLKRAMB0-4 中读取 4 个输入特征图的 8 行数据总计 32 行数据,传输给 CU 阵列的 4 个 RELU 单元进行激活运算,并将 4 个计算后的输出特征图存回到 BLKRAMB0-4 的对应地址,循环计算 $M/4$ 次后得到 M 个激活后的输出特征图。

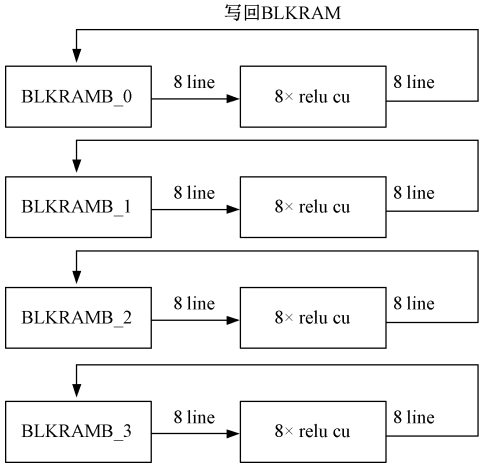


图 5　单次激活运算结构及数据流向

Fig. 5　Activation computing architecture and data flow

CNP 的池化层运算与激活层运算类似,同时读取 4 个输入特征图进行池化计算后将结果存回原输入特征图的 BLKRAM 中。

在进行 BLKRAM 设计时,将 BLKRAMA 和 BLKRAMB 设计成全对称结构,使子 BLKRAM 的位宽、容量和数量完全相同,所以它们都可以用来存储输入特征图或者输出特征图,以此种对称结构排列特征图后,可以通过对调数据流向来实现多层神经网络的计算,并且能达到最大效率的数据复用效果。比如第一层网络计算中,BLKRAMA 作为输入层,将数据传输给 CU 阵列计算后,把本层输出结果存储在 BLKRAMB 中,在计算第 2 层网络时,调换存储与计算顺序,将 BLKRAMB 作为输入层,把第 1 层计算结果读取出来送给 CU 阵列计算后,输出结果存储在 BLKRAMA 中,以同样的方法计算第 3 层及之后多层网络,如图 6 所示。通过此种 BLKRAM 对称结构和对调数据流向的方法可以实现多层网络完全运行在 SRAM 内,并且使得 CNN 计算效率达到最大。

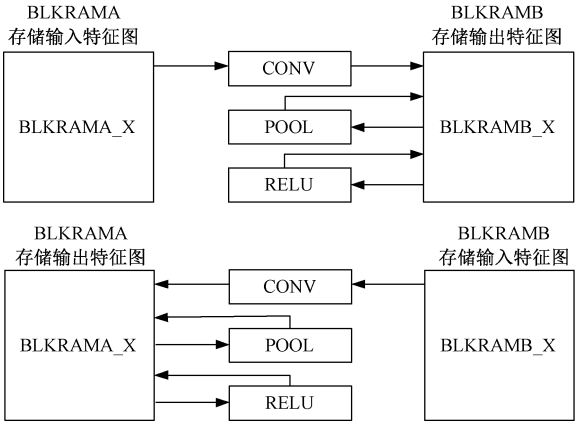


图 6　BLKRAM 双向数据流动

Fig. 6　BLKRAM bi-direction data flow

2.4　特征图读取与写出

由于网络模型的差异及每层特征图数量及尺寸的不同,并且受限于电路面积影响(SRAM 不能太大),BLKRAM 不可能完全存储下所有的特征图以及中间计算结果,这就需要在网络计算时将特征图在 BLKRAM 及 DDR 之间进行传输,而 BLKRAM 和 DDR 之间的数据传输效率直接影响到网络计算效率,为了最大化提高传输速度,本文设计了特征图读取及写出指令,并且支持特征

图的裁剪与拼接操作,使得任意尺寸特征图都可以高效率的进行神经网络运算。

如图 7 所示,假设 N 个输入特征图(IN_CH0-IN_CHN)的大小超过了 BLKRAM 的容量,可以对输入特征图进行裁剪,如裁剪出 IN_CH0_CUT-IN_CHN_CUT,这样裁剪后 BLKRAM 就可以存储所有 N 个输入特征图,在特征图读入指令中,通过配置 IN_CH0-IN_CHN 的尺寸(src_width x src_height)和特征图 DDR 中地址,以及 IN_CH0_CUT-IN_CHN_CUT(cut_width x cut_height)和裁剪图的首地址,CNP 在运行特征图读入指令时,BUS_CTRL 可以从 DDR 中自动读取 IN_CH0_CUT-IN_CHN_CUT 再将其按照 BLKRAM 的排列规则将其写入 BLKRAM 中,并且可以根据所读取的实时图像坐标发出连续读请求到总线上,完成特征图从 DDR 到 BLKRAM 最大效率搬移。特征图写出的操作过程与读取类似,把计算完成的输出特征图从 BLKRAM 中读取到 DDR 中。

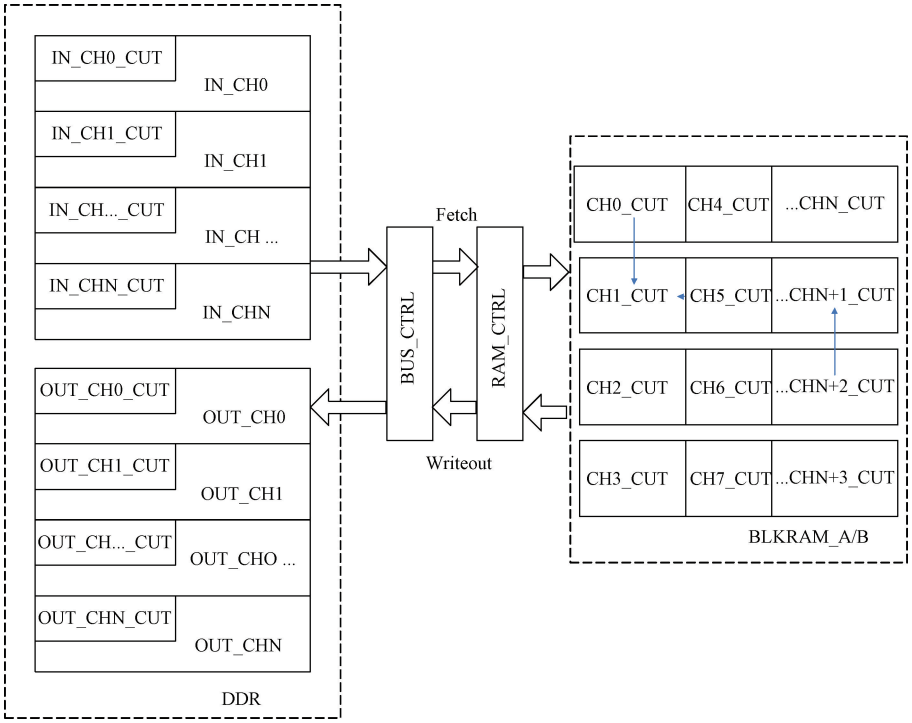


图 7 特征图读入及写出功能示意图

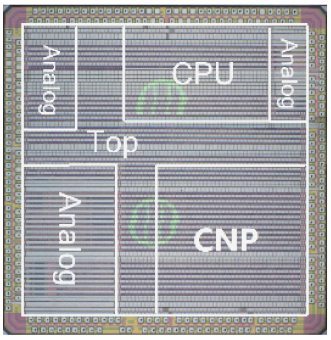
Fig. 7 Diagram of feature map read and write

上述的特征图读入和写出指令可以配合其它运算指令,实现输入特征图任意尺寸任意数量的裁剪,进行神经网络计算后,再把输出特征图进行任意尺寸及任意数量的拼接,不用受限于输入特征图及输出特征图的尺寸及数量要求,由于是 CNP 指令支持硬件自动读取及写入,所以不需要任何 CPU 及 DMA 进行数据搬移及处理,并且针对特征图的特点及 BLKRAM 排列规则进行专门的电路设计,其数据搬移效率远远大过 DMA 传送速度。

以 tiny yolo v2 的神经网络模型为例,第一层输入特征图分辨率为 224×224, 特征图数量为 3,特征图占用内存容量为 224×224×3×2(16 bit)= 301 kB,本芯片采用的 BLKRAM 总容量为 256 kB,BLKRAMA/BLKRAMB 各占用 128 kB,BLKRAMA 无法存储第一层的全部输入特征图,所以需要对输入特征图在 DDR 中进行图像分割(平均分为 4 块),将分割后图像分别读取到 BLKRAMA 中进行计算,再将计算后放入 BLKRAMB 中的图像读取到 DDR 中拼接成完整的输出特征图,其后第 2 层至第 6 层都需要对特征图进行裁剪、拼接及数据搬移操作。本文对比采用 DMA 搬移及上述硬件读写特征图两种传输方式,使用 DMA 进行特征图搬移时,由于拼接和裁剪需要 CPU 频繁计算起始地址及长度,使 DMA 无法高效率连续读写,导致运行此网络时在数据搬移上耗时 30 ms 左右,而使用硬件读写特征图操作仅耗时 10 ms 左右,提升了近 3 倍效率,并且不消耗任何 CPU 及 DMA 资源。

3 结果分析与讨论

以下是芯片的测试结果与分析。图 8 是该 SoC 芯片的顶层解剖图以及工艺信息。芯片通过中心国际的 40 nm LL 工艺设计完成。CNP 最高速率达到 500 MHz,最低运行功耗为 45.3 mW。在 standby 时,整个加速器处于电源关闭状态。整个 SoC 待机功耗为 3 μA。



Technology	SMIC 40 nm LL
CNP Area	1.04
Data Length	16 bit/8 bits
ACC width	32 bits
Total SRAM	262 KB
Supply Voltage	0.99 ~1.21 V
Core Frequency	48 ~500 MHz
Peak Performance	288 GOPS
Energy Efficiency	3.22 TOPS/W
Power Consumption	89.4 mW
Layer Size	ALL
Layer Type	CNN
Area Efficiency	190 GOPS/mm^2

图 8　芯片顶层图以及性能概要

Fig. 8　Chip top layout and performance summary

图 9 是实验用的工程开发板,通过这块开发板,可以完成该 SoC 绝大部分功能。PC 通过调试接口连接 SoC 芯片进行调试。板卡上配备 SDRAM 芯片支撑片外存储,Norflash 芯片保存 CPU 运行程序,LCD 接口驱动显示,USB/eMMC/NAND 支持大容量数据存储,以及 IIC,UART 等接口。

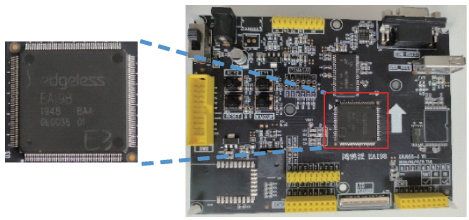


图 9　芯片封装图以及系统板卡

Fig. 9　Chip package and system demo board

在 CNN 典型的应用中有人脸识别,人形识别,如图 10 所示为运用 tiny-yolo 模型完成人形检测及 mobilenet 完成人脸识别的典型应用,在芯片上运行 tiny-yolo/mobilenet 模型可达到 30 frames/s 的速度,实现了实时人体检测及人脸识别。这仅仅是芯片人工智能应用场景的一部分,在语音、手势、物体的识别上也能发挥很好的功效。

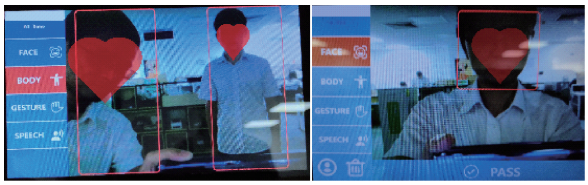


图 10　两种典型的人工智能应用

Fig. 10　Two classic AI applications

表 1 列出了与近两年业内深度学习的论文进行数据对比。从对比表中看出,虽然其中的算力对比中并不出色。但从最后两行的数据来看,在功耗与面积方面有明显的优势,领先业内的近期的数据。在智慧家庭等应用

表 1　与已有文献性能对比

Table 1　Performance comparison with existing references

对比项目	文献[11] (ISSCC)	文献[12] (JSSC)	文献[13] (ISSCC)	文献[14] (JSSC)	文献[15]	本文所提出的设计
流片与否	是	是	是	是	否	是
工艺制程/nm	28	40	40	65	65	40
面积/mm^2	1.87	2.4	121.6	16	4.35	1.514
工作电压/V	0.65~1.05	0.55~1.1	1.1	0.63~1.1	1.0	0.99~1.21
PE 精度/bit	4, 8, 16	1~16	1~4	1~16	16	8, 16
最大性能/GOPS	76 (16b)	102(16)	1960 (4b)	345.6(16b)	281(16b)	288(16b)
工作频率/MHz	200	204	330	200	650	500
功耗/mW	300 (1.05 V)	288 (1.1 V)	3300(1.1 V)	297 (1.1 V)	859(1.0 V)	89.4 (1.1 V)
功耗效率/(TOPS/W)	0.53(16b)	0.3	0.59(4b)	3.08(16b)	0.282(16b)	3.22(16b)
面积效率/(GOPS/mm^2)	40.6(16b)	42.5(1b)	16.1(16b)	21.6(16b)	64.6(16b)	95.1(16b)

中并不是拼绝对算力,而是拼功耗与算力效率。从结果可以看出,在嵌入式应用领域,本文提出的方案具有较高的竞争力。

4 结 论

本文通过对典型卷积神经网络加速器的创新结构设计;通过对 SRAM 的高效运用,双 SRAM 模块进行组合运用,与卷积层的输入与输出相对应。把数据与神经网络相关数据运算结合起来,实现 MAC 单元的高效率计算,并且极大限度减少 DDR 读写,达到运算存储一体化。并通过流片后的测试数据反映出:功耗效率,性能面积效率在业内领先。在嵌入式领域,特别在智慧家庭的智能化方向,降低了使用门槛,同时也能扩展到其它嵌入式领域应用。

参考文献

[1] 刘勤让,刘崇阳. 利用参数稀疏性的卷积神经网络计算优化及其 FPGA 加速器设计[J]. 电子与信息学报,2018,40(6):1368-1369.
LIU Q R, LIU CH R. Calculation optimization for convolutional neural networks and FPGA-based accelerator design using the parameters sparsity [J]. Journal of Electronics & Information Technology, 2018, 40(6):1368-1369.

[2] 岳颀,马彩文. 指数弹性动量卷积神经网络及其在行人检测中的应用[J]. 哈尔滨工业大学学报,2017,49(5): 159-164.
YUE X, MA C W. A deep convolution neural network for object detection based[J]. Journal of Harbin Institute of Technology, 2017,49(5):159-164.

[3] 郭继昌,郭昊,郭春乐. 多尺度卷积神经网络的单幅图像去雨方法[J]. 哈尔滨工业大学学报,2018,50(3): 185.
GUO J CH, GUO H,GUO CH L. Single image rain removal based on multi-scale convolutional neural network [J]. Journal of Harbin Institute of Technology, 2018,50(3): 185.

[4] 海金. 神经网络与机器学习[M]. 北京: 机械工业出版社, 2017: 77-138.
HAYKIN S. Neural networks and learning macines, third edition[M]. Beijing: China Machine Press, 2017:77-138.

[5] LE C Y, BENGIO Y, HINTON G. Deep learning[J]. Nature, 2015, 521(7553), 436-436.

[6] 徐欣,刘强,王少军. 一种高度并行的卷积神经网络加速器设计方法[J]. 哈尔滨工业大学学报, 2020, 52(4): 31-37.
XU X, LIU Q, WANG SH J. A highly parallel design method for convolutional neural networks accelerator[J]. Journal of Harbin Institute of Technology, 2020, 52(4): 31-37.

[7] 王 巍,周凯利,王伊昌,等. 基于快速滤波算法的卷积神经网络加速器设计[J]. 电子与信息学报,2019, 41(11):2578-2580.
WANG W, ZHOU K L, WANG Y CH, et al. Design of convolutional neural networks accelerator based on fast filter algorithm[J]. Journal of Electronics & Information Technology, 2019,41(11):2578-2580.

[8] LIN C H, CHENG C C, TSAI Y M, et al. A 3.4-to-13.3TOPS/W 3.6TOPS dual-core deep-learning accelerator for versatile AI applications in 7 nm 5G smartphone SoC [C]. IEEE Int. Solid-State Circuits Conf. San Francis-Co, USA, 2020: 134-134.

[9] 吴飞. 人工智能导论:模型与算法[M]. 北京: 高等教育出版社, 2020: 214-221.
WU F. Introduction to artificial intelligeme: Models and algorithms[M]. Beijing: Higher Education Press, 2020: 214-221.

[10] 李鼎基,糜泽羽,吴保东,等. 基于跨虚拟机零下陷通信的加速器虚拟化框架[J]. 软件学报, 2020,1(10): 3019-3037.
LI D J, MI Z Y, WU B D, et al. Accelerator virtualization framework based on inter-VM exitless communication[J]. Journal of Software, 2020,31(10): 3019-3037.

[11] MOONS B, UYTTERHOEVEN R, DEHAENE W, et al. Envision: A 0.26-to-10 TOPS/W subword-parallel dynamic-voltage-accuracy frequency-scalable convolutional neural network processor 28 nm FDSOI[C]. IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, 2017: 246-257.

[12] MOONS B, VERHELST M. A 0.3- 2.6 TOPS/W precision-scalable processor for real-time large-scale ConvNets[C]. Proc. IEEE Symp. VLSI Circuits (VLSI-Circuits), 2016: 1-2.

[13] UEYOSHI K, ANDO K, HIROSE K, et al. QUEST: A 7.49 TOPS multi-purpose log-quantized DNN inference engine stacked on 96 MB 3D SRAM using inductive

coupling technology in 40 nm CMOS[C]. IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, 2018: 216-218.

[14] LEE J, KIM C, KANG S, et al. UNPU: An energy-efficient deep neural network accelerator with fully variable weight bit precision[J]. IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig, 2020: 173-185.

[15] 赵博雅. 基于卷积神经网络的硬件加速器设计及实现研究[D]. 哈尔滨. 哈尔滨工业大学, 2020.
ZHAO B Y. Study on design and implementation of hardware accelerators based on convolutional neural networks[D]. Harbin: Harbin Institute of Technology, 2020.

作者简介

易冬柏,2005 年于华中科技大学获得学士学位,2007 年于华中科技大学获得硕士学位,现为浙江大学博士研究生,主要研究方向为微电子,集成电路方向。
E-mail: 11931086@zju.edu.cn

Yi Dongbai received his B. Sc. degree and M. Sc. degree both from Huazhong University of Science and Technology in 2005 and 2007, respectively. He is currently a Ph. D. candidate at Zhejiang University. His main research directions include microelectronics and integrated circuit.

陈恒(通信作者),2008 年于华南理工大学获得学士学位,2011 年于华南理工大学获得硕士学位,现为零边界集成电路有限公司工程师,主要研究方向为神经网络处理器架构与电路设计,数字 SoC 电路设计。
E-mail: daniel.chen@cn.gree.com

Chen Heng (Corresponding author) received his B. Sc. degree and M. Sc. degree both from South China University of Technology in 2008 and 2011, respectively. He is currently a senior engineer in Zhuhai edgeless semiconductor Co., Ltd., his main research interests include the architecture of neural network and digital SoC design.