An NoC-based CNN Accelerator for Edge Computing

Jianing Gao, Qiming Shao, Fangyu Deng, Qin Wang, Naifeng Jing, Jianfei Jiang*
Department of Microelectronics and Nanoscience, Shanghai Jiao Tong University, Shanghai, China
*Email: Jiangjianfei@sjtu.edu.cn

2023 IEEE 15th International Conference on ASIC (ASICON) | 979-8-3503-1298-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/ASICON58565.2023.10396346

***Abstract*—Nowadays, convolutional neural networks (CNN) are developing in the direction of low-computation and high-precision lightweight algorithms. However, conventional accelerators cannot achieve high performance for lightweight algorithms. Network-on-Chip (NoC) can solve this problem because of its configurable on-chip dataflow and large on-chip bandwidth. In this paper, we propose a new dual-layer NoC-based CNN accelerator with an optimized router for edge computing. Furthermore, we design a loop unfolding and mapping scheme based on the NoC so that the accelerator architecture can adapt to the data delivery requirements of various CNN algorithms with strong algorithmic generality. Our design is based on the SMIC 40nm process, and the core area is 9 mm2. The experimental results on MobileNet and RepVGG show a peak performance of 51.2 GOPS. The power consumption is 301.7 mW, and the energy efficiency reaches 169.7 GOPS/W at 200 MHz.**

Keywords—Convolutional Neural Networks, Network on Chip, Accelerator

I. INTRODUCTION

Convolutional neural networks have shown tremendous feature extraction capability and image classification accuracy in many domains like face recognition, automatic driving, medical diagnosis. Nowadays, deploying CNN training on the cloud and then deploying CNN inference on edge devices has become a mainstream solution. Due to edge devices' resource limitations, mainstream AI algorithms like CNN are gradually developing towards lightweight to improve inference efficiency and accuracy [1]. Depth-wise (DW) convolution [2] is commonly used in this kind of network, which has quite a different dataflow from ordinary convolution, and conventional CNN accelerators cannot efficiently compute DW convolution. Therefore, an accelerator for lightweight algorithms is essential.

Many CNN accelerators have emerged, such as Eyeriss [3], Gemmini [4], Eyeriss v2 [5], and Envision [6]. Eyeriss proposed Row-Stationary (RS) dataflow for the first time and used the multicast NoC structure, which supports deploying multiple algorithms. But its data delivery is easy to block, and the actual computational efficiency and energy efficiency are low. Eyeriss v2 transmits the data through multi-layer NoC interconnection. However, it still has the disadvantage of low energy efficiency, resource overhead, and difficulty in meeting off-chip memory access requirements. Envision achieves energy efficiency up to 10 TOPS/W with dynamic voltage accuracy frequency scaling (DVAFS) technology. Gemmini is a full-stack DNN accelerator design infrastructure to enable systematic evaluation of deep learning architectures. However, running MobileNetV2 with 256 PEs at 1 GHz, only a processing speed of 18.7 frames per second could be achieved.

Lightweight CNN algorithms are iteratively updated; fixed data interconnect designs such as systolic array cannot deploy new algorithms efficiently. However, the NoC architecture can separate the memory access dataflow from the computation dataflow, reducing the complexity of the data communication on CNN models. In addition, in edge devices, the on-chip memory and the computing units are greatly limited, which makes it hard to provide sufficient bandwidth and excellent peak computational performance. On the other hand, the power consumption of both memory access and computing needs to be strictly controlled.

For the problems mentioned above, in this paper, we propose a dual-layer NoC-based accelerator for lightweight CNN networks, which can achieve high computational efficiency. The rest of this paper is structured as follows. Section II proposes the hardware design of our accelerator. Section III describes the computation dataflow and hardware mapping in detail. Section IV presents the performance evaluation of our design compared with existing designs. Finally, the conclusion of this paper is drawn in Section V.

II. HARDWARE ACCELERATION

A. Architecture of Accelerator

A dual-layer NoC-based architecture is proposed, as shown in Fig. 1, to support lightweight CNN processing. The design includes four parts: 1)NoC array; 2)PE clusters; 3)Loop controller; and 4)Instruction decoder.

1) NoC array: The NoC array delivers data in different patterns to support calculation. To get a trade-off between area cost, MAC density, and bandwidth, the size of the NoC array is set to 4×4. We implement two layers to separate weight and feature map(fmap) delivery, which address the problem of data congestion and deadlock. Weight delivery layer: The bottom layer shown in Fig. 1 comprises 4 linear NoCs, supporting weight delivery in the column direction. On-chip memory access is performed by the 4 nodes at the top. Fmap delivery layer: The top layer performs a 4×4 mesh structure. The 4 nodes on the left are input nodes, and the bottom nodes are output nodes. The routers at the same position on the two layers are connected to the same PE cluster.

2) Instruction decoder: To support various dataflow for CNN algorithms, we have designed a dedicated Instruction Set Architecture (ISA) as the control core of the accelerator, including system instruction, memory access instruction, routing configuration instruction, and ALU calculation instruction. The accelerator will load the specific scheme of the CNN processing into on-chip INST MEM at the start. The decoder sends the decoded instructions to the loop control module and packagers to complete the configurations.

3) Loop controller: The loop controller controls the memory access interface, input data packagers, and weight packagers. The off-chip memory access interface is AXI4 with a bit width of 256. The packagers split 256-bit data into four 64-bit flits and add headers and trailers containing information such as the destination router, the PE cluster

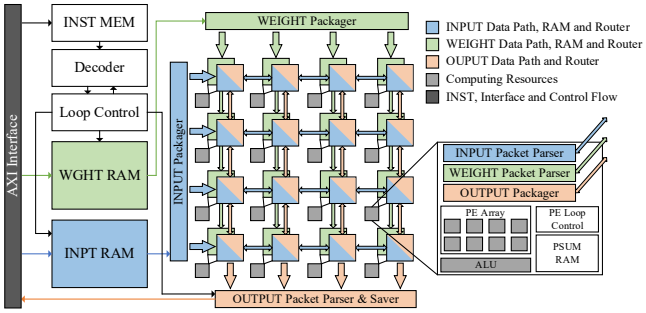

Fig. 1. Architecture overview

computation strategy, the arithmetic and logic unit (ALU) configuration, and the loop dimensions. And then 64-bit flits are passed into the specific PE array and split into 8 parallel PEs.

4) PE cluster: Each PE cluster includes 8 processing elements, and each PE contains 1 MAC operation. PE array's structure is described in the following subsection in detail.

B. Structure of PE Array

As is shown in Fig. 2, the PE array consists of two input data parsers, an output data packager module, a PE loop control module, a partial sum memory, a PE computing unit, and an ALU. Each PE contains a multiple-and-accumulate (MAC) operation that can process 8-bit fixed-point inputs and weights with 8-bit variable precision. The result with 16-bit variable precision can represent 32-bit fixed-point data. Additionally, the adder's input must be aligned with the decimal point location to calculate accurately. So the signed bit-width expander completes the bit-width expansion based on precision configuration information in the packet header.

After a single accumulation is completed, the bit-width cutter cuts 32-bit data into 16-bit variable-precision fixed-point PSUM and then saves it to PSUM RAM. The final 128-bit accumulation result generated by 8 PE is transferred to the ALU, where the bit-width cutter cuts them into 64-bit data. The cutter can also choose whether to perform the ReLU6 based on the configuration. Then, add PSUM with bias, and ALU will select bypass, max pooling, or average pooling according to the configuration. Finally, the packaging module transfers the 64-bit output data to the off-chip mem through the NoC.

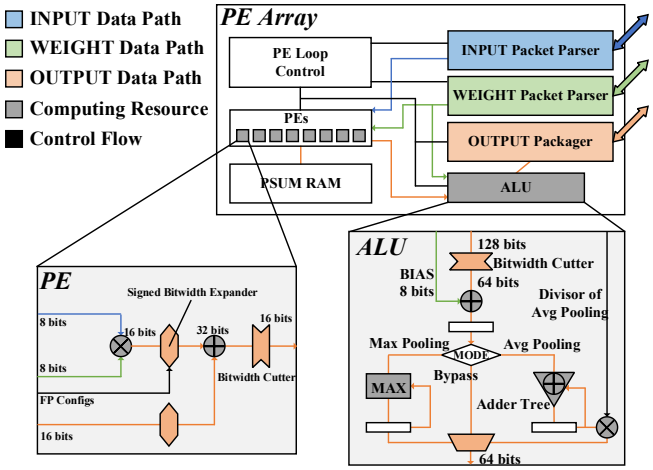

Fig. 2. Structure of PE array

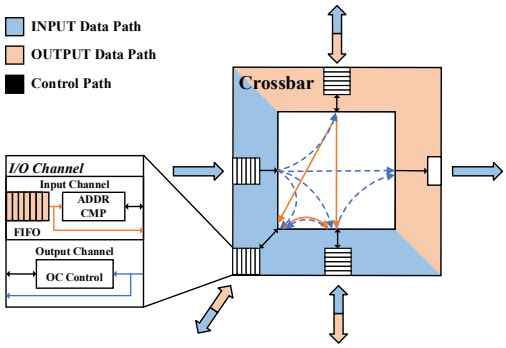

Fig. 3. Hardware architecture of router

C. Optimization of Router

Due to the dual-layer NoC routing input data and weight separately, the data flow of each layer is fixed. As a result, to minimize power consumption, we optimized the router structure by cutting unnecessary dataflow directions in the top mesh layer, as shown in Fig. 3. Three input and output channels are reserved: up, down, and local directions. And in the left-right direction, input data delivery from left to right is reserved, resulting in only the input channel on the left and the output channel on the right. Physical design results show that the optimized router reduces the NoC array's logic resources and power consumption by 43.1% and 40.9%, respectively.

III. Computation Dataflow and Hardware Mapping

A. Computation Dataflow

A flexible and efficient computation dataflow is of great significance for various lightweight CNN algorithms. This section provides a detailed introduction to the dataflow for different algorithms based on the dual-layer NoC architecture proposed above. Moreover, to optimize the NoC load, we propose a scheme for loop unfolding in configurable dimensions. TABLE I lists the dimension parameters, where $I\times$ denotes the input data, $O\times$, and $W\times$ are the output data and weight, $\times C$, $\times W$, and $\times H$ denote the number of channels, the data width, and data height. We adopt the Row Stationary Dataflow [3] in the PE due to its remarkable energy efficiency. Each PE is responsible for computing multiple input and output channels, denoted as $ICpe$ and $OCpe$.

Fig. 4 depicts the data flow inside a single PE; multiple input channels are shown in the same color. Additionally, the weights will perform a sliding window operation according to the stride, allowing weight reuse. So the PE first computes the data in the $OCpe$ dimension, followed by the $OWpe$ dimension. PSUMs are stored in the local memory of the node. The proposed dataflow is also applicable to FC layers.

In the DW CONV layer, each kernel performs independent convolution operations on input channels, so the output channel per PE cluster ($OCpe$) is 1. To enhance parallelism, as depicted in Fig. 5, we add the parallel dimension of $OHpe$, representing the number of output map's rows processed by each PE. According to the characteristics of DW convolution, the value of $OHpe$ should satisfy:

$$OHpe \le min\{WH, IH - WH + 1\} \quad (1)$$

After the multiple weight sliding window shifts and calculations are completed, one PE can generate four data in different rows of the same column in the output map.

TABLE I. DIMENSION PARAMETERS OF CNN LAYERS

Input	IC, IW, IH
Weight	OC, IC, WW, WH
Output	OC, OW, OH

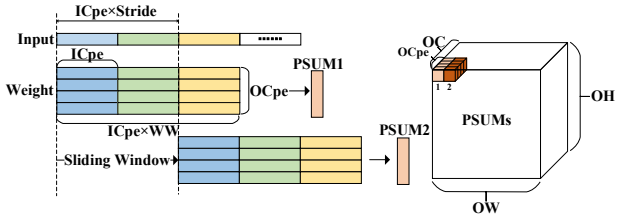

Fig. 4. Dataflow of convolution

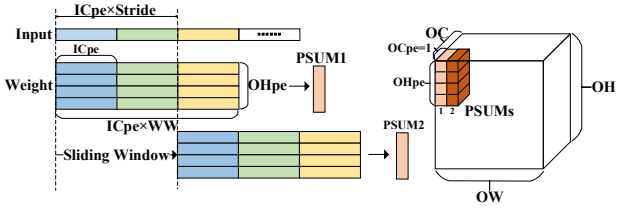

Fig. 5. Computation dataflow of DW convolution

B. Hardware Mapping

To adapt to the data delivery requirements of various CNN algorithms, we propose a loop unfolding scheme and mapping scheme for our NoC-based accelerator with strong algorithmic generality.

As is shown in TABLE II, we split four dimensions of the original loop: IC, OC, OW, and OH. After being unfolded, the 6-layer-loop turns into 13 layers. And the detailed process is presented in Algorithm 1. The 13-layer-loop can be divided into five parts, identified in the comments. Each part contains multiple loops arranged in descending order of data reuse intensity from inside to outside.

The innermost part is inside one PE unit, consistent with the RS data flow introduced above. Next is the PE cluster loop, where OWp represents the 8 PEs computing different positions of the same output row in parallel. Then comes the NoC-router loop, $\{OCx, OCy\}$ represent parameters in both horizontal and vertical directions of NoC, which means each NoC node will compute different output channels to achieve parallel computation. The next outer layer is the on-chip mem access loop, and the outermost layer is the off-chip access loop.

The DW convolution is partially different from the CONV, mainly in the DW CONV using $OHpe$ instead of $OCpe$ to improve the computation/access latency ratio. The FC layer is similar to the CONV, with the difference that the PE cluster loop dimension is set to OC to achieve full parallel computation at the PE level.

Different layers adopt different routing modes based on data reuse. For the CONV layer, each node computes different output channels in parallel, resulting in reusing the input data but not the weight. Therefore, at the NoC level, input data is multicast while weight is unicast. Within a PE cluster, due to the parallelism of weight in the OW dimension, input data is unicast and weight is multicast.

In the FC layer, weight reuse is challenging, and only the input data is spatially reusable. Therefore, weight is unicast at both the NoC and PE levels, while input data is multicast. The DW CONV lacks spatial reuse of input data, but weight can be temporally reused. Therefore, input data is unicast. With the increased depth of the CNN algorithm, the length and width of input data in the last few layers become smaller, limiting weight reuse. Consequently, multicast mode is employed only at the PE level.

Algorithm 1 Convolution Loop Split

Input: INPUT: an array of input feature maps
Input: WEIGHT: an array of convolutional kernels
Output: OUTPUT: an array of output feature maps

```
// Off-chip Mem;
for ocs=1 to OCs do
  for ohs=1 to OHs do
    for wh=1 to WHs do
      for ics=1 to ICs do
        for ohp=1 to OHp do
          // On-chip Mem;
          for ows=1 to OWs do
            // Router;
            for ocy=1 to OCy do
              for ocx=1 to OCx do
                // PE Cluster;
                for owp=1 to OWp do
                  // PE;
                  for owpe=1 to OWpe do
                    for ocpe=1 to OCpe do
                      for ww=1 to WW do
                        for icpe=1 to ICpe do
                          ow=ows×OWp×OWpe+owp×OWpe+owpe;
                          oh=ohs×OHp+ohp;
                          ic=ics×ICpe+icpe;
                          iw=ow+ww;
                          ih=oh+wh;
                          oc=ocs×OCy×OCx×OCpe+ocy×OCx×
                          OCpe+ocx×OCpe+ocpe;
                          OUTPUT [oc][ow][oh]+=INPUT [ic][iw][ih]
                          ×WEIGHT [oc][ic][ww] [wh];
```

TABLE II. THE FORMULA FOR LOOP UNFOLDING

Dimension	Loop Unfloding Scheme
IC	$ICs \times ICpe$
OC	$OCs \times OCy \times OCx \times OCpe$
OW	$OWs \times OWy \times OWx \times OWp \times OWpe$
OH	$OHs \times OHp$

IV. RESULTS AND DISCUSSION

We built the neural network algorithm deployed by our accelerator using TensorFlow frameworks. Furthermore, we fused batch normalization and convolution to reduce computational and access overhead. Additionally, we utilized TensorFlow Lite to complete the training and inference of the fixed-point quantization, with a quantization bit width of 8 bits. We set the quantization precision of the input data in the input layer to 8 bits decimal. Additionally, the quantization precision of the input and output data in other layers is set to 3 bits integer and 5 bits decimal, and the weight is quantized using a dynamic precision scheme. Finally, our quantization scheme ensures similar and acceptable accuracy loss (less than 0.8% Top1 accuracy loss compared with the pre-trained model). Moreover, we developed a framework in Python to generate instruction profiles based on the parameters and structure of the network.

A. Layout of Accelerator

The accelerator layout is shown in Fig. 6. Our design is implemented in SMIC 40nm process and has a chip area of 9mm2 with a total of 68 KB SRAM. Additionally, our accelerator operates at a frequency of 200 MHz with a voltage of 1.1 V, consuming an average power of 301.7 mW. Moreover, the peak computational performance of our

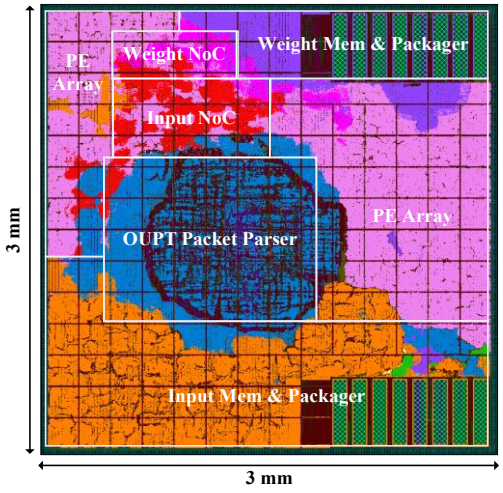

Fig. 6. Layout of accelerator

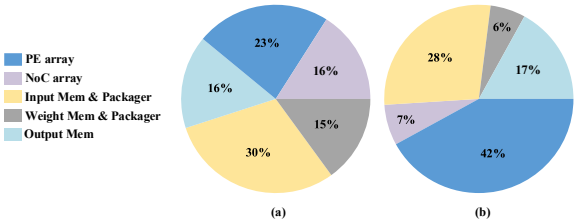

Fig. 7. Power and area breakdown: (a) Power Breakdown (b) Area breakdown

accelerator is 51.2 GOPS, with an energy efficiency of 169.7 GOPS/W.

We plot the area and power breakdown from the design data, which is shown in Fig. 7. In the proposed accelerator, the PE array occupies the largest area due to the inclusion of a 2KB SRAM for storing partial sums. The NoC array consumes only 16% of total power and occupies 7% of the total area, which indicates that the NoC array can achieve a highly versatile on-chip network with less resource and power consumption, making it an energy-efficient architecture.

B. Performance Evaluation

We demonstrate our design's effectiveness by deploying MobileNetV1, MobileNetV2, and RepVGG, which have distinct algorithm structures, calculations, and memory accesses. Our design achieves 38.2 GOPS in MobileNetV1 and 31.2 GOPS in MobileNetV2. Additionally, RepVGG attains high computational performance in our accelerator, achieving 47.4 GOPS.

TABLE III shows the comparison between our design and the state-of-the-art design. Eyeriss only supports CONV operation, but our accelerator can deploy multiple algorithms and perform better in terms of area, energy efficiency, and computational efficiency. Moreover, our design achieves 2.2× throughput improvement over Eyeriss [3] for the same convolution deployment.

Eyeriss v2 is an improved design of the Eyeriss. But with an aggregated external read and write bandwidth of 25600MB/s, the throughput of Eyeriss v2 running MobileNet will decrease by 24% [5]. According to the above, we calculate the area and energy efficient data of Eyeriss v2 in TABLE III. Our accelerator's energy efficiency is 15.3% higher than that of Eyeriss v2, while achieving a 54% power and 70% area reduction, making it better suited for low-power and low-resource edge devices. Compared to Gemmini, the actual computation performance of the MobileNetV2 deployment in this paper is 2.8× higher.

TABLE III. COMPARISON OF STATE-OF-THE-ART ACCELERATORS

	[3]	[4]	[5]	Our design		
Tech (nm)	65	28	65	40		
Frequency (MHz)	200	1000	200	200		
Area (mm2)	12.25	1.3	28*	9		
SRAM (KB)	181.5	256	246	68		
Peak throughput (GOPS)	67.2	512	153.6	51.2		
Average power (mW)	236	-	650.9	301.7		
Energy Efficiency (GOPS/W)	90.7	-	147.2*	169.7		
CONV	yes	yes	yes	yes		
FC	no	yes	yes	yes		
Lightweight algorithm	no	yes	yes	yes		
Algorithm deployment	CONV	V2	V1	V1	V2	Rep VGG
Actual performance (GOPS)	21.4	11.09	95.8*	38.2	31.2	47.4

V. CONCLUSION

Recent attention has been drawn to energy-efficient CNN accelerators for lightweight algorithms, particularly in edge devices. In this paper, we propose a dual-layer NoC-based CNN accelerator with an optimized router for edge computing. We also design a loop unfolding and mapping scheme for various algorithms. The accelerator has been designed based on SMIC 40nm process and has an area of 9mm2. Our accelerator can achieve a peak computational performance of 51.2 GOPS at 200 MHz. The power consumption is 301.7 mW and the energy efficiency reaches 169.7 GOPS/W.

REFERENCES

[1] M. Capra, B. Bussolino, A. Marchisio, M. Shafique, G. Masera, and M. Martina, "An updated survey of efficient hardware architectures for accelerating deep convolutional neural networks," Future Internet, vol. 12, no. 7, pp. 113-134, 2020.

[2] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1800–1807.

[3] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," IEEE Journal of Solid-State Circuits, vol. 52, no. 1, pp. 127–138, 2017.

[4] H. Genc, S. Kim, A. Amid, A. Haj-Ali, V. Iyer, P. Prakash, J. Zhao, D. Grubb, H. Liew, H. Mao et al., "Gemmini: Enabling systematic deep-learning architecture evaluation via full-stack integration," in 2021 58th ACM/IEEE Design Automation Conference (DAC), 2021, pp. 769–774.

[5] Y.-H. Chen, T.-J. Yang, J. Emer, and V. Sze, "Eyeriss v2: A flexible accelerator for emerging deep neural networks on mobile devices," IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 9, no. 2, pp. 292–308, 2019.

[6] B. Moons, R. Uytterhoeven, W. Dehaene, and M. Verhelst, "14.5 Envision: A 0.26-to-10tops/w subword-parallel dynamic-voltage-accuracy-frequency-scalable convolutional neural network processor in 28nm fdsoi," in 2017 IEEE International Solid-State Circuits Conference (ISSCC), 2017, pp. 246-247.