A Precision-Scalable Energy-Efficient Bit-Split-and-Combination Vector Systolic Accelerator for NAS-Optimized DNNs on Edge

Kai Li[1], Junzhuo Zhou[1], Yuhang Wang[1], Junyi Luo[1], Zhengke Yang[1], Shuxin Yang[1], Wei Mao[1], Mingqiang Huang[1] and Hao Yu[1]

[1]*School of Microelectronics, Southern University of Science and Technology*, Shenzhen, China

***Abstract*—Optimized model and energy-efficient hardware are both required for deep neural networks (DNNs) in edge-computing area. Neural architecture search (NAS) methods are employed for DNN model optimization with resulted multi-precision networks. Previous works have proposed low-precision-combination (LPC) and high-precision-split (HPS) methods for multi-precision networks, which are not energy-efficient for precision-scalable vector implementation. In this paper, a bit-split-and-combination (BSC) based vector systolic accelerator is developed for a precision-scalable energy-efficient convolution on edge. The maximum energy efficiency of the proposed BSC vector processing element (PE) is up to 1.95× higher in 2-bit, 4-bit and 8-bit operations when compared with LPC and HPS PEs. Further with NAS optimized multi-precision CNN networks, the averaged energy efficiency of the proposed vector systolic BSC PE array achieves up to 2.18× higher in 2-bit, 4-bit and 8-bit operations than that of LPC and HPS PE arrays.**

***Index Terms*—Precision-scalable, bit-split-and-combination, vector systolic, accelerator, CNN, NAS**

I. Introduction

Deep learning on edge has received considerable attention due to a low-carbon realization of AI [1]. The essential challenge is that convolutional neutral network (CNN) models for deep learning are becoming more complicated with larger number of parameters, which can lead to enormous power or carbon consumption [2]. Neural Architecture Search (NAS) can optimize CNN network models with multi-precision (or bit-width) layers [3]. Compared with single-precision networks, multi-precision networks are more suitable for efficient data processing on edge due to their lightweight model size with maintained accuracy [4].

As shown in Fig. 1, a typical NAS flow results in that the precision of each DNN layer is optimized with selected bit-width by NAS algorithm, or a so-called multi-precision network. However, there is no hardware designed that can efficiently support a high performance multi-precision computation, especially on edge devices that have limited hardware resources [5].

With bit-serial operation, several state-of-the-art multi-precision accelerators with 1-bit to 16-bit fully-variable weight bit-precision have been proposed previously [6]–[9]. However, serial-inner-product method in these works can only support multi-precision weight input. The registers for cyclically usage occupy large hardware area cost. As shown in Fig. 2(a), the bottom-up low-precision-combination (LPC) method combines the low-precision units with configurable shifters to achieve multi-precision MAC operations in one clock cycle, such as BitFusion [10], BitBlade [11] and other structures [12], [13], [14]. Moreover, in Fig. 2(b), the design is based on a top-down high-precision-split (HPS) method by using the whole partial products of the high-precision units with gated area to obtain the multi-precision operations, such as subword parallel structure [15]. However, neither LPC nor HPS methods are precision scalable for both throughput and power consumption, e.g. energy efficiency.

Several NAS-based multi-precision network VGG-16 models

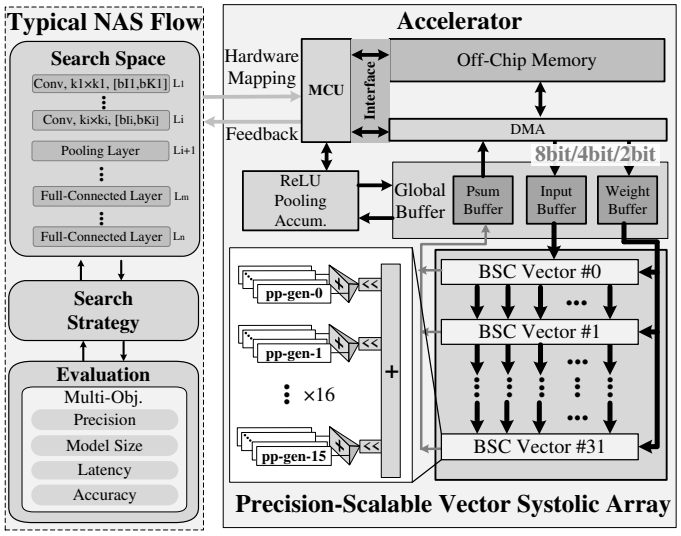

Fig. 1. Typical NAS flow with proposed precision-scalable vector systolic accelerator.

TABLE I
Evaluated NAS-Based Multi-Precision CNN Benchmarks.

CNN	Dataset	Model Weights	--- Proportion of --- 8-bits	4-bits	2-bits
VGG-16	CIFAR-10	138.0 MBytes	10.2%	89.8%	0%
LeNet-5	MNIST	0.5 MBytes	0%	55.0%	45.0%
ResNet-18	ImageNet	13.0 MBytes	5.5%	94.5%	0%
NAS-Based	-	-	21.8%	58.6%	19.6%

Note: NAS-Based summarized several VGG-16 models trained by NAS

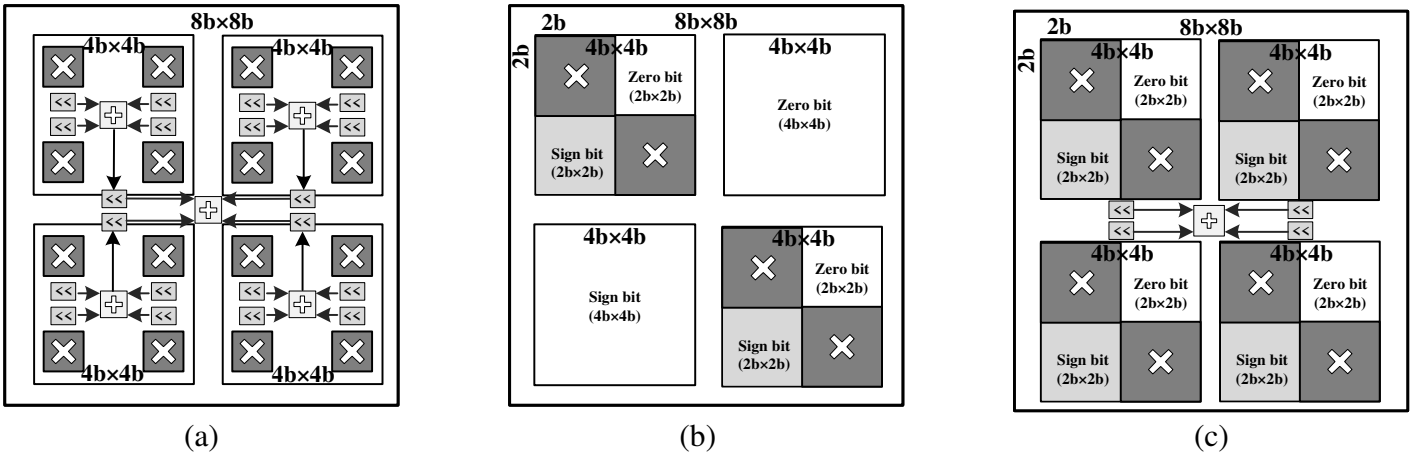

Fig. 2. Different methods to implement precision-scalable MAC: (a) LPC method, (b) HPS medthod, (c) proposed BSC method.

and other multi-precision CNN benchmark are analyzed and summarized by the Table I [16], [17]. The results show that 4-bit operations account for more than 50% of the total computation. Therefore, a novel multi-precision vector operation unit based on BSC is proposed, which takes 4-bit as the basic operation unit. As shown in Fig. 1 as well as Fig. 2(c), a precision-scalable energy-efficient BSC vector systolic accelerator is proposed in this paper for processing NAS-optimized multi-precision CNN on edge. Compared with LPC and HPS methods, the BSC method in this paper can achieve precision scalable energy efficiency during multi-precision computation. What is more, a vector BSC processing element (PE) together with its vector systolic PE array are both developed to further improve the precision scalable energy efficiency. The experiments show that the maximum energy efficiency of the proposed BSC vector PE is up to 1.95× higher in 2-bit, 4-bit and 8-bit operations when compared with LPC and HPS PEs. Further with NAS optimized multi-precision CNN networks, the averaged energy efficiency of the proposed vector systolic BSC PE array achieves up to 2.18× higher in 2-bit, 4-bit and 8-bit operations than that of LPC and HPS PE arrays.

II. Previous Work

For the design of conventional multi-precision MAC units, LPC and HPS are two commonly used methods.

A. LPC Method

The LPC MAC unit achieves multi-precision MAC operations by adding low-precision multiplication units with configurable-shifters based on the bottom-up strategy [10].

Fig. 2(a) shows the LPC design for 8-bit, 4-bit and 2-bit modes, respectively. Each LPC MAC unit includes 16 basic 2-bit signed multiplication. These 16 results are divided into 4 groups and are added together respectively to get 4 internal sums. The values of the shifters in Fig. 2(a) are different in various modes.

For 2-bit mode, 16 results and 4 internal sums are added up sequentially without shifting. For 4-bit mode, the results in each group are added up with internal shifting values setting to 0, 2, 2 and 4 bits. Then four internal sums are added up without shifting. For 8-bit mode, internal sums are generated in the same way as 4-bit mode. Then the internal sums are added up with global shifting values setting to 0, 4, 4 and 8 bits. The LPC MAC unit in 2-bit, 4-bit and 8-bit modes can execute 16, 4, and 1 MAC operations in one clock cycle, respectively. However, fully reconfigurability of this MAC unit requires large hardware cost, which makes low energy efficiency for higher-precision operations.

B. HPS Method

The HPS MAC unit achieves multiple-precision MAC operations by splitting one high-precision multiplier into multiple lower-precision multipliers by gating logic blocks and signed expand, which is based on the top-down strategy [15].

Fig. 2(b) shows the HPS design. A HPS MAC unit is a two-dimensional symmetrical scalable architecture. In 2-bit, 4-bit or 8-bit mode, the HPS MAC unit can execute one 8bit×8bit, two 4bit×4bit, or four 2bit×2bit operations, respectively. At different bit-width modes, the corresponding partial products have similar shifting values, and these partial products will be added up for the final MAC results. Such configuration of implicitly performing the addition through the multiplier array cells reduces the usage of additional adders, which enable the addition of these products.

However, to maintain the same input bandwidth, it is necessary to sacrifice its hardware utilization when precision scales down. So a low hardware utilization rate is inevitable when HPS executes low-precision modes. Hardware utilization rate of such configuration are 100% for 8-bit mode, 50% for 4-bit mode, and only 25% for 2-bit mode, as the Fig. 2(b) shown.

III. BSC Multi-Precision Vector MAC

To solve the problems of high energy consumption of LPC MAC and low hardware utilization of HPS MAC, BSC MAC is proposed in this paper by considering the energy efficiency and hardware resource utilization. Compared with HPS and

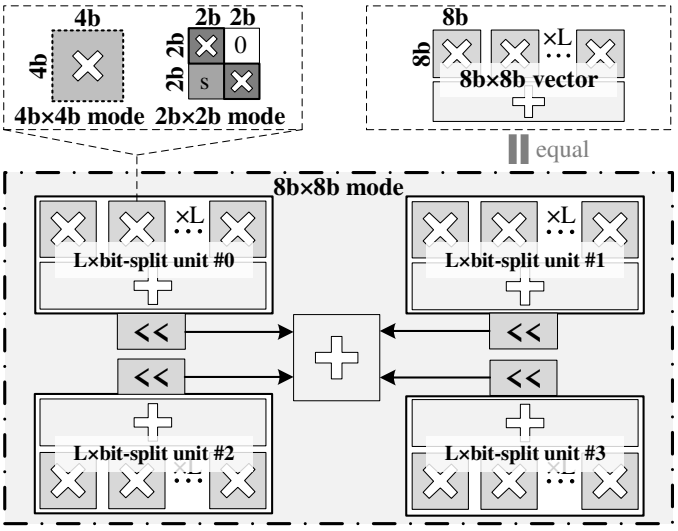

Fig. 3. Implementation of BSC vector MAC with length L.

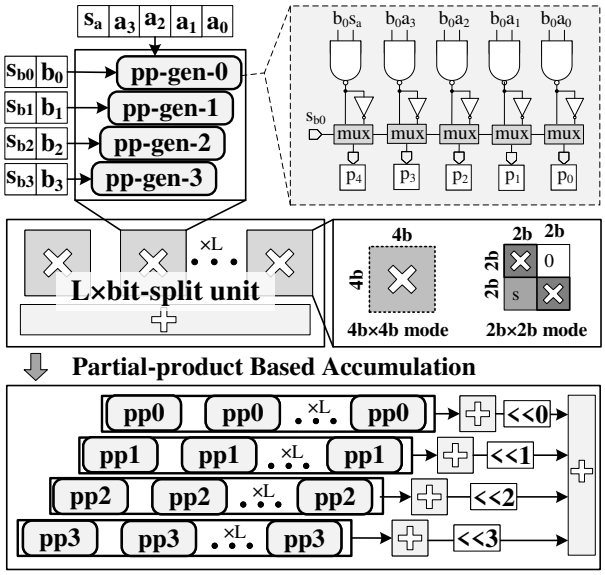

Fig. 4. Bit-split unit implementation with same shift partial-product accumulation.

LPC methods, the proposed BSC method improves the energy-efficient performance further.

A. BSC Method Analysis

The MAC method of BSC is shown in Fig. 2(c). It mainly consists of four 4-bit bit-split units, four shifters and an adder. An signed 4-bit bit-split unit is designed to support one signed 4bit×4bit and two signed 2bit×2bit operations. Then composability of the signed 4-bit bit-split unit is used to implement 8bit×8bit operation. For the bit-split unit, two 2bit×2bit operations are implemented by gated and signed expand. The 8bit×8bit operation result is gained from the addition of partial sums after shifting 0, 4, 4 and 8 bits left, while the result of 4bit×4bit operation or 2bit×2bit operation is gained from the addition of the partial sums without shifting.

Compare with the 3-bit BitBrick (a signed 3-bit multiplier) used in LPC method for sined 2bit×2bit operation, the signed 4-bit bit-split unit will have an advantage in both throughput and power consumption for 2bit×2bit operations. On the other hand, BSC method will double the throughput of 4bit×4bit and 2bit×2bit operation when compared with HPS method.

B. Vector Implementation

To further improve MAC throughput and energy efficiency, BSC vector is proposed. As shown in Fig. 3, there are four bit-split units with length L in the BSC vector. The bit-split unit is the basic unit that constitutes the BSC MAC. Each of the bit-split unit support one 4bit×4bit or two 2bit×2bit operation.

The outputs of those bit-split units are sent to the shifter to get the partial sum. The 8bit×8bit vector operation result is gained from the addition of partial sum after shifting 0, 4, 4 and 8 bits left, while the result of 4bit×4bit vector operation or 2bit×2bit vector operation is gained from the addition of the partial-sum without shifting.

In addition, the bit-split unit with length L is designed elaborately. As shown in Fig. 4, each of the bit-split unit is implemented by 4 partial-product generators (pp-gen 0 to 3). Note that the partial product generated in bit-split unit do not add up directly. The partial products with the same shifting values in different bit-split units are accumulated together. As Fig. 4 shows, after the partial products with the same shifting values are added up, the sum results are shifted by 0, 1, 2 and 3 bits and added up.

C. Signed Implementation

To realize signed multi-precision operation, the generation process of signed/unsigned partial-product is shown in Fig. 4. S_a and S_{bx} are used to indicate whether the input a and b are signed, S_a and S_{b0} are shown as an example in the Fig. 4. Note that all inputs are in 2-complement format. S_a and $a_{3\sim0}$ are formed a 5-bit input, which is used to implement signed 4bit×4bit or unsigned 4bit×4bit in 8bit×8bit operation. S_{b0} and S_a are used to generate signed output to avoid the increment after the inversion. In addition, the signed adder tree will add up the $S_{b0}\cap S_a$ and the 5-bit output with other pp-gen output in the same bitwise.

As Fig. 4 shows, partial product is firstly generated by the NAND logic, which is based on whether the input b_x (b_0 as an example) is signed (S_{b0} is 1'b1). Then the multiplexer will select the output of the NAND. In the meantime, if the b_0 is 1'b1, the $S_{b0}\cap S_a$ will be 1'b1, which is used to avoid the increment caused by the inverse. If the input b_0 is unsigned, S_{b0} will be 1'b0, then the multiplexer will select the output of the NOT and $S_{b0}\cap S_a$ will be 1'b0.

IV. Precision-Scalable Vector Systolic PE Array

Systolic array architecture has been introduced in current neural network accelerators, which can improve energy efficiency greatly by utilizing data reuse and reducing data-driving power [18], [19]. Google tensor processing unit (TPU) adopts the weight-stationary systolic array [18]. The weights are fixed in each PE while the partial-sum and the input-data

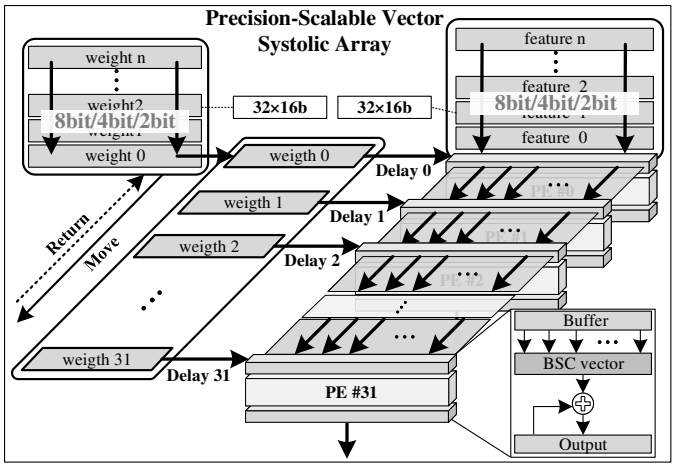

Fig. 5. BSC precision-scalable vector systolic PE array dataflow.

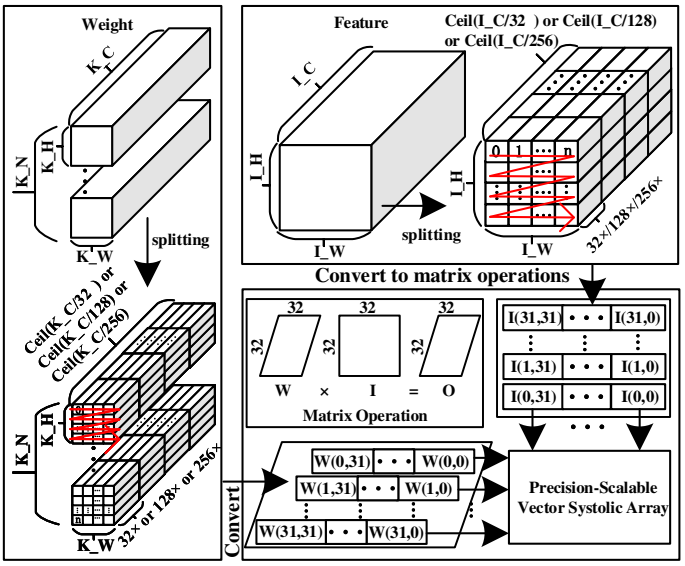

Fig. 6. Convolution and matrix mapping in vector systolic array.

are delivered horizontally and vertically in the PE array. With the weight-stationary method, the partial sums and the input data are reused in the PE array to improve the energy efficient performance. This paper proposed the precision-saclable vector systolic PE array, which is aimed to be more suitable for proposed BSC vector.

A. Precision-Scalable Vector Systolic PE Array

The proposed BSC vector is basic unit of the precision-scalable vector systolic PE array. Fig. 5 shows the dataflow of the precision-scalable vector systolic PE array. 32 PEs are used to build the vector systolic array. As Fig. 5 shows, each PE includes input buffer, BSC vector and output buffer. By considering the hardware cost and bandwidth limitations, the proposed BSC vector with length of 32 is set as the basic unit of the PE. Based on the BSC vector, the precision-scalable vector systolic PE array with 32 PEs supports 1024 8bit×8bit, 4096 4bit×4bit or 8192 2bit×2bit operations. The dataflow of the precision-scalable vector systolic PE array will be transmitted as vector with length 32. Note that each element of the vector has a bitwidth of 16 bits for BSC-based vector, while the LPC-based and HPS-based vector are 32 bits and 8bits, respectively.

As Fig. 5 shows, the feature data will be transmitted by the buffer of PE0, while the weight data will be directly sent to the buffer of PE0 to PE 31 after delay 0 clock to 31 clocks, respectively. At the 1st posedge of the clock, the 1st data of feature is sent to the buffer of PE0. The 1st data of weight is also sent to the buffer of PE0. Then at the 2nd posedge of the clock, the 1st data of feature is transmitted to the buffer of PE1 from the buffer of PE0. At the same time, the 2nd data of feature is sent to the buffer of PE0 to replace the 1st data of feature. The 1st data of weight stays the same at the buffer of PE0 and the 2nd data of weight is sent to the buffer of PE1. The output buffer of PE0 is generated as 1st result. At each subsequent posedge of the clock, the feature data from the previous PE buffer will be transmitted to the next PE buffer until it reaches the last PE31. The output buffer of previous PE will generate result. The weight data will be sent to the next PE buffer as the increasing number of clock. After 32 clocks, the weight data will be sent to all the 32 buffers of the PE array, which means 32 clocks in a cycle.

B. Convolution and Matrix Mapping

The proposed BSC precision-scalable vector systolic PE array support vector operation with length 32 and has 32 PEs in parallel. To implement convolution operation, the data of weight and feature are split according to the hardware resources of the proposed vector systolic PE array.

As Fig. 6 shows, weight data is W[K_N][K_C][K_W][K_H] and feature data is I[I_C][I_W][I_H]. In the proposed design, the vector length is 32, 128 and 256 in 8-bit, 4-bit and 2-bit operation, respectively. The vector length is parallel to the channel direction of K_C and I_C. Therefore, K_C and I_C will be split to 32, 128 and 256 in 8-bit, 4-bit and 2-bit operation, respectively. For the K_N direction, the number of PE is parallel to K_N. Then the K_N will be be splitting split to 32. For the order of operations after the convolution is split, the direction of I_W and K_W will be calculated first and then is the direction of I W and K W and the direction of K_H and I_H are calculated sequentially, which is shown in Fig. 6. Convolution operation will be convert to matrix operation, which is a common practice of systolic array [18], [19].

For the matrix operation, Fig. 6 shows an example of 32×32 matrix operation. Assume I and W are both 32×32 matrixes. As the subsection IV. A introduced, the data of I is sent to the precision-scalable vector systolic array as feature data and W is sent as weight data. The output data will form an output matrix O in the form of systolic.

V. Experiment Results

A. Methodology

1) 28nm+dc+ptpx+vcs: The aforementioned vector units designs in RTL are synthesized by Synopsys Design Compiler

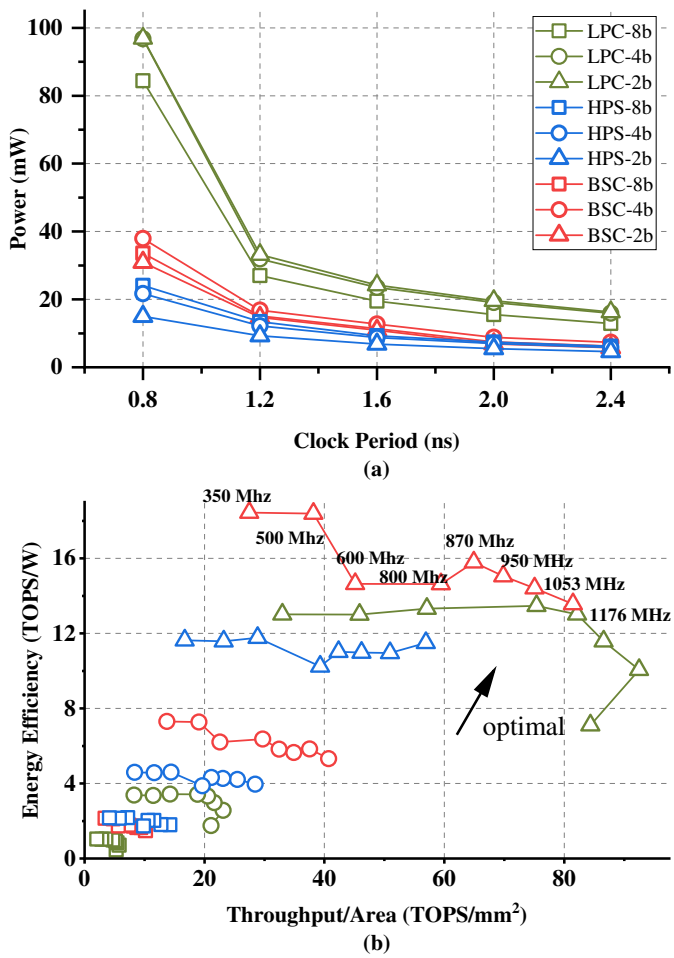

Fig. 7. Precision scalability comparison among BSC, LPC and HPS: (a) Energy vs. Delay; (b) Energy efficienc y vs. Area efficiency

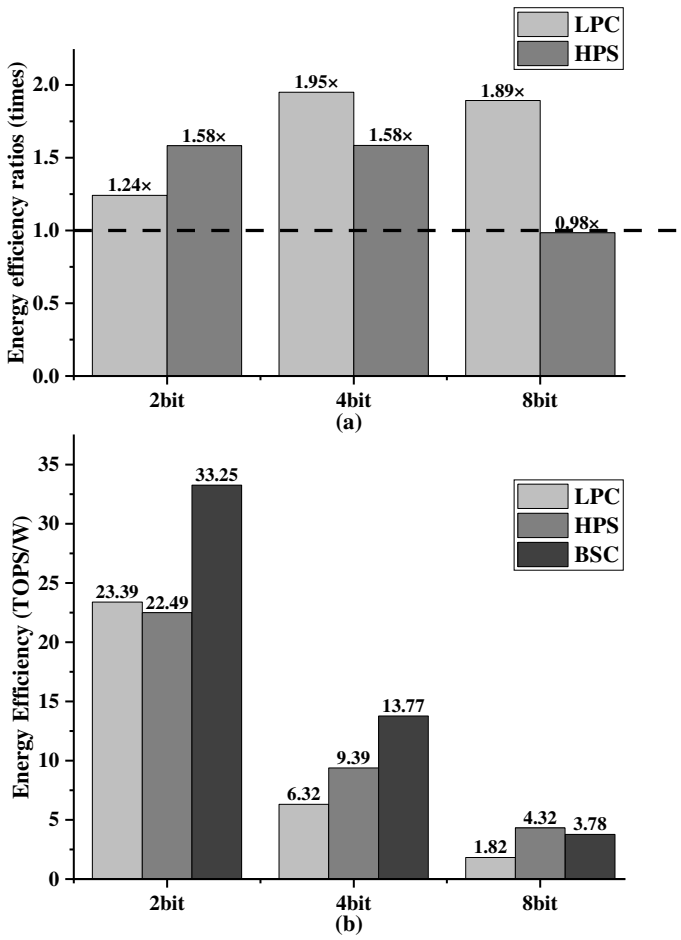

Fig. 8. Energy efficiency comparison of BSC, LPC and HPS: (a) Precision-scalable vector; (b) Vector systolic PE array

and implemented under SIMC 28-nm process. The Synopsys Design Compiler and PrimeTime PX are used to obtain the power, performance and area (PPA) values. VCS tools are applied to verify the functionality and achieve 100% functional coverage in different bit-width operations modes.

2) Baseline of LPC: For the experiments with LPC, we reproduce the design of BitBlade [11] and eliminate the design of asymmetric bitwidths (2bit×4bit and 4bit×8bit). For the sake of fairness, we implement LPC vector with length L=32 for LPC designs.

3) Baseline of HPS: For the experiments with HPS, we reproduce subword-parallel design [15] and eliminate the design of asymmetric bitwidths (2bit×4bit and 4bit×8bit). For the sake of fairness, we implement HPS vector with length L=32 for HPS designs.

B. Comparison of Precision Scalablility

As Fig. 7 shows, the clock period is set up from 0.8ns to 2.4ns and all timing reports indicate that no timing slack. The three designs are all synthesized with vector length L=32 and synthesized by Synopsys Design Compiler with different frequency. Then PrimeTime PX are used to obtain the power.

The excitation of simulation includes three different operation mode (8bit×8bit, 4bit×4bit and 2bit×2bit) settings, and the power consumption of three different modes is also simulated. All the results are shown as Fig. 7(a). It is obvious that BSC method has better performance and in power compared with LPC. The power of BSC method in 2b-mode is 50% lower than that of LPC method at clock frequency equal to 500 MHz.

To fully evaluate the three designs, the energy efficiency vs area efficiency curves of specific precision modes is shown as Fig. 7(b). From Fig. 7(b), comparing with LPC and HPS, the BSC vector has an advantage in energy efficiency in 2-bit and 4-bit. Fig. 7(b) shows that under the condition of the same throughput, BSC method has better energy efficiency. Under the condition of the same energy efficiency, BSC method has better throughput.

C. Comparison of Max Energy Efficiency

In Fig. 8, the energy efficiency ratios of BSC vector to other works and the performance of all the vector systolic array in three modes are provided.

As shown in Fig. 8(a), comparing maximum energy efficiency with state-of-the-art works, BSC performs better energy efficiency in most of the cases. Compared with LPC, BSC is

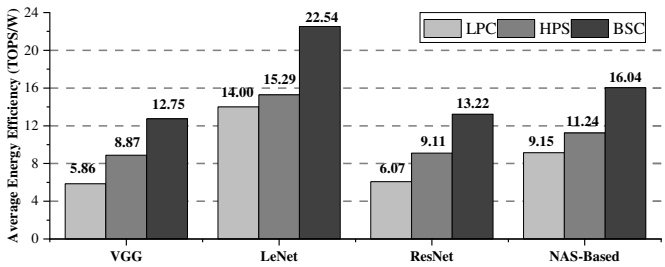

Fig. 9. Average energy efficiencies of precision-scalable vector systolic PE array with HPS, LPC and BSC vectors on multi-precision CNN benchmarks.

around 1.24× better in 2-bit mode and about 2× as good as LPC in 4-bit and 8-bit modes. Compared with HPS, energy efficiency of BSC is around 1.6 × superior than that of HPS in 2-bit and 4-bit modes.

The energy efficiency of vector systolic PE array with LPC, HPS and BSC vector is summarized in Fig. 8(b). The vector systolic PE array with BSC vector achieves the maximum energy efficiency of 33.25 TOPS/W and 13.77 TOPS/W in 2-bit and 4-bit mode, respectively.

D. Comparison of Multi-Precision Computation on NAS-CNNs

Fig. 9 presents the energy efficiency performance of the accelerators on multiple-precision CNN benchmarks of Table I [16], [17]. The values represent the energy efficiency of the proposed BSC accelerator and the other published works. Further with NAS optimized multi-precision CNN networks, the average energy efficiency of the proposed vector systolic BSC PE array achieves up to 12.75 TOPS/W, 22.54 TOPS/W, 13.22 TOPS/W ans 16.04 TOPS/Win NAS optimized multi-precision VGG-16, LeNet-5, ResNet-18 and NAS-Based benchmarks, which is 2.17×, 1.61×, 2.18× and 1.75× higher than that of LPC PE array, and 1.43×, 1.47×, 1.45× and 1.43× higher than that of HPS PE array. The largest energy efficiency ratio 2.18× is achieved, which benefits from the vector systolic architecture and high energy-efficient multi-precision BSC method in 2-bit and 4-bit operation.

VI. Conclusion

In this paper, a BSC vector systolic accelerator with improved energy-efficient performance is proposed. The maximum energy efficiency of the proposed BSC vector PE is up to 1.95× higher in 2-bit, 4-bit and 8-bit operations when compared with LPC and HPS PEs. Further with NAS optimized multi-precision CNN networks, the maximum average energy efficiency of the proposed vector systolic BSC PE array achieves up to 22.54 TOPS/W in NAS optimized multi-precision LeNet-5, and the maximum improvement of average energy efficiency is 2.18× higher than that of LPC PE array and HPS PE array.

VII. Acknowledgement

This work is supported by the Shenzhen Science and Technology Program (Grant No. KQTD20200820113051096), Innovative Team Program of Education Department of Guangdong Province (Grant No.2018KCXTD028), the Key-Area Research and Development Program of Guangdong Province (Grant No. 2019B010142001), National Key R&D Program of the Ministry of science and technology (Grant No. 2021YFE0204000), and National Natural Science Foundation of China (Grant No. 62034007).

References

[1] X. Xu and *et al.*, "Scaling for edge inference of deep neural networks," *Nature Electronics*, vol. 1, no. 4, pp. 216–222, 2018.

[2] P. Gysel and *et al.*, "Ristretto: A framework for empirical study of resource-efficient inference in convolutional neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 11, pp. 5784–5789, 2018.

[3] K. Wang and *et al.*, "Haq: Hardware-aware automated quantization with mixed precision," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 8604–8612.

[4] B. Wu and *et al.*, "FBNet: Hardware-aware efficient convnet design via differentiable neural architecture search," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 10 726–10 734.

[5] V. Camus and *et al.*, "Review and benchmarking of precision-scalable multiply-accumulate unit architectures for embedded neural-network processing," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 4, pp. 697–711, 2019.

[6] P. Judd and *et al.*, "Stripes: Bit-serial deep neural network computing," in *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 2016, pp. 1–12.

[7] A. Delmas and *et al.*, "Tartan: Accelerating fully-connected and convolutional layers in deep learning networks by exploiting numerical precision variability," *arXiv preprint arXiv:1707.09068*, 2017.

[8] S. Sharify and *et al.*, "Loom: Exploiting weight and activation precisions to accelerate convolutional neural networks," in *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*. IEEE, 2018, pp. 1–6.

[9] J. Lee and *et al.*, "Unpu: An energy-efficient deep neural network accelerator with fully variable weight bit precision," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 1, pp. 173–185, 2018.

[10] H. Sharma and *et al.*, "Bit fusion: Bit-level dynamically composable architecture for accelerating deep neural network," in *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*, 2018, pp. 764–775.

[11] S. Ryu and *et al.*, "Bitblade: Area and energy-efficient precision-scalable neural network accelerator with bitwise summation," in *Proceedings of the 56th Annual Design Automation Conference 2019*, 2019, pp. 1–6.

[12] S. Ghodrati and *et al.*, "Bit-parallel vector composability for neural acceleration," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2020, pp. 1–6.

[13] S. Ryu and *et al.*, "A 44.1 tops/w precision-scalable accelerator for quantized neural networks in 28nm cmos," in *2020 IEEE Custom Integrated Circuits Conference (CICC)*. IEEE, 2020, pp. 1–4.

[14] W. Liu and *et al.*, "A precision-scalable energy-efficient convolutional neural network accelerator," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 10, pp. 3484–3497, 2020.

[15] L. Mei and *et al.*, "Sub-word parallel precision-scalable mac engines for efficient embedded dnn inference," in *2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*. IEEE, 2019, pp. 6–10.

[16] I. Hubara and *et al.*, "Quantized neural networks: Training neural networks with low precision weights and activations," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6869–6898, 2017.

[17] B. Zhuang and *et al.*, "Towards effective low-bitwidth convolutional neural networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7920–7928.

[18] N. P. Jouppi and *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA)*, 2017, pp. 1–12.

[19] Y.-H. Lai and *et al.*, "SuSy: A programming model for productive construction of high-performance systolic arrays on fpgas," in *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2020, pp. 1–9.