9.3 A 40nm 4.81TFLOPS/W 8b Floating-Point Training Processor for Non-Sparse Neural Networks Using Shared Exponent Bias and 24-Way Fused Multiply-Add Tree

Jeongwoo Park*, Sunwoo Lee*, Dongsuk Jeon

Seoul National University, Seoul, Korea

*Equally Credited Authors (ECAs)

Recent works on mobile deep-learning processors have presented designs that exploit sparsity [2, 3], which is commonly found in various neural networks. However, due to the shift in the machine learning community towards using non-sparse activation functions such as Leaky ReLU or Swish for better training convergence, state-of-the-art models no longer exhibit the sparsity found in conventional ReLU-based models (Fig. 9.3.1, top). Moreover, contrary to error-tolerant image classification tasks, more difficult tasks such as image super-resolution require higher precision than plain 8b integers not just for training, but for inference without large accuracy degradation (Fig. 9.3.1, bottom). These changes offer new challenges faced by mobile deep-learning processors: they must process non-sparse networks efficiently and maintain higher precision for more challenging tasks.

To overcome such challenges, this paper presents a deep learning processor with support for efficient inference and training for various modern neural networks, which advances energy and area efficiency, while achieving robust end-to-end training. Our key contributions are: (1) an 8b floating point data format with shared exponent bias (FP8-SEB) for robust training in low precision, (2) a processing architecture employing 24-way Fused Multiply-Add (FMA) trees and high-precision accumulators that improves training accuracy and energy efficiency, and (3) a 2D routing scheme on input/output points of the processing element array for flexible training of various models with minimal hardware overhead. Based on these contributions, our neural-network training processor achieves 4.81TFLOPS/W energy efficiency, 567GFLOPS performance, and robust training on various models including Generative Adversarial Network (GAN), Long-Short-Term-Memory (LSTM), transformer models, as well as Convolutional Neural Network (CNN).

Figure 9.3.2 shows the proposed FP8-SEB format suitable for training various models from scratch to match the accuracy of full-precision training. While prior works [1-3] also employed a FP8 format for training, they were limited to using mixed precision of FP8/FP16 [2, 3] or suffered from training accuracy degradation [1, 2]. We represent the elements of a tensor in 1-4-3 (1b sign, 4b exponent, and 3b mantissa) FP8 format, while each tensor contains an extra byte to express the exponent bias shared across all the elements in the tensor. The cost of additional bias is negligible since it is shared across a large number of elements in a tensor; for instance, the shared biases only occupy 0.003% of memory in ResNet-18 training. Since all the elements in each tensor share the same exponent bias, tensor multiplication can be performed in FP8, while the shared biases of two-operand tensors are combined separately through simple addition/subtraction (Fig. 9.3.2, top right). The bias of a tensor is also updated after each graph computation by simply incrementing or decrementing the bias by 1 for overflow/underutilization. We find that the numerical precision of accumulation is crucial in training, as suggested in [1]. For instance, 8b accumulation suffers accuracy loss on CIFAR-10 (5% for LeNet) or simply fails to converge for ResNet-18. Consequently, we conservatively use FP30 accumulation in 1-6-23 format. This ensures training convergence without accuracy loss on not just ImageNet training, but also more complicated tasks. In experiments, using naïve FP16 on GAN (super-resolution) and LSTM (image captioning) training fails to converge, whereas our training scheme shows no performance degradation (Fig. 9.3.2, bottom right). The final accumulation result is quantized back into FP8, and hence all the inputs/outputs of the processor are still represented in 8b, minimizing costly external memory accesses.

The overall architecture of the processor consists of a 4×16 processing element array, 2D routing at the input/output points of the processing elements, two buffers with 40kB and 192kB each, and a mixed-precision 16-lane vector processor (Fig. 9.3.3). Each processing element contains the main processing logic employing a 24-way FMA tree, a high-precision accumulator, and a 960B of scratchpad memory for storing accumulated values. The on-chip buffers are split into two categories: a high-performance buffer with a prefetcher for maximum performance (40kB) and a high-capacity buffer for area efficiency (192kB). The mixed-precision vector processor supports various operations including linear vector operation in FP8/FP16 mixed precision and element-wise FP32 arithmetic operations for end-to-end neural network training.

To implement the FP8 training scheme in a high-precision and energy-efficient manner, we adopt an N-way FMA tree, depicted in Fig. 9.3.4. A conventional MAC unit constantly loses information when adding multiplication results into much larger partial sums. Conversely, an N-way FMA scheme combines multiple multiplication results in lossless representations before adding to the partial sum, suppressing rounding errors inherent in floating point representations. In a toy example of 1024×1024 matrix multiplication, conventional MAC (1-way FMA) shows a PSNR of 14.3dB, whereas 32-way FMA shows a PSNR of 24.1dB (Fig. 9.3.4, bottom left). We choose our design point to be 24-way for balancing the flexibility of adder trees and precision. It should be noted that these improvements do not come at the cost of more logic or energy; the energy consumption reduces from 15.9pJ/MAC (1-way FMA) to 1.9pJ/MAC (24-way FMA) since combining partial products from multipliers using a single adder tree provides more space for logic optimization and the cost of accessing the accumulation memory is amortized over 24 MAC operations.

Figure 9.3.5 describes the routing strategy for supporting both training and inference of CNN/FC/RNN efficiently with N-way FMA trees. By deploying routing only on inputs/outputs of the processing arrays instead of on intermediary results, routing resources consume just 0.34% of the total area and 0.72% of the total energy. The routing flexibility is sufficient for different types of convolution-like computations required in training; routing on input points are used for the convolution feedforward stage and matrix multiplications that require the result to be accumulated inwards; routing on output points are used for the convolution feedbackward stage and deconvolution layers that require the result to be accumulated outwards. For instance, in the feedforward stage of convolutional layers, pixel values of different channels are read from high-performance buffers. These pixel values then go through the global input routing unit, consisting of a 3-stage channel-selection mux and shift registers, which respectively align data according to their channels and spatial locations (Fig. 9.3.5, bottom left). In the feedbackward stage of convolution layers, final accumulated values from the processing elements are aligned spatially by being added to the time-delayed accumulated values of neighboring pixels (Fig. 9.3.5, bottom right).

Figure 9.3.6 compares with prior work [2-6]. Our training scheme better matches the full-precision accuracy of 69.6% with 69.0%, compared to 67.4% in [1] (our implementation, 67.3% in the original paper) and 68.2% in [2] for ResNet-18 training in simulation using a bit-accurate model of the processor (Fig. 9.3.6, top left). Note that we used GPU-based simulation to extract training accuracy up to 90 epochs (= 90M images) in a reasonable amount of time, and the fabricated chip was confirmed to exactly match the simulation model in measurements. Fabricated in 40nm LP CMOS, the processor is measured to consume 13.1mW at 0.75V, 20MHz with the maximum energy efficiency of 4.81TFLOPS/W, and 230mW at 1.1V, 180MHz with the maximum performance of 567GFLOPS and the area efficiency of 90.7GFLOPS/mm^2. The processor exhibits 43.0% fewer external memory accesses compared to the design in [2] for the same ResNet-18 training due to the 8b representation for communication with external memory (Fig. 9.3.6, top right). In addition, the processor provides 2.48× higher energy efficiency for training non-sparse models by using the FP8-SEB format combined with 24-way FMA trees, high-precision accumulators, and flexible 2D routing. The die photograph and chip details are shown in Fig. 9.3.7.

Acknowledgement:
This work was supported by the National Research Foundation of Korea (Grant No. NRF-2019R1C1C1004927), the KIST Institutional Program (Project No. 2E30610-20-058), and the IC Design Education Center (IDEC).

References:
[1] N. Wang et al., “Training Deep Neural Networks with 8-bit Floating Point Numbers,” *NeurIPS*, 2018.
[2] J. Lee et al., “LNPU: A 25.3TFLOPS/W Sparse Deep-Neural-Network Learning Processor with Fine-Grained Mixed Precision of FP8-FP16,” *ISSCC*, pp. 142-143, 2019.
[3] S. Kang et al., “GANPU: A 135TFLOPS/W Multi-DNN Training Processor for GANs with Speculative Dual-Sparsity Exploitation,” *ISSCC*, pp. 140-141, 2020.
[4] C. Kim et al., “A 2.1TFLOPS/W Mobile Deep RL Accelerator with Transposable PE Array and Experience Compression,” *ISSCC*, pp. 136-137, 2019.
[5] B. Fleischer et al., “A Scalable Multi-TeraOPS Deep Learning Processor Core for AI Training and Inference,” *IEEE Symp. VLSI Circuits*, pp. 35-36, 2018.
[6] J. Oh et al., “A 3.0 TFLOPS 0.62V Scalable Processor Core for High Compute Utilization AI Training and Inference,” *IEEE Symp. VLSI Circuits*, 2020.

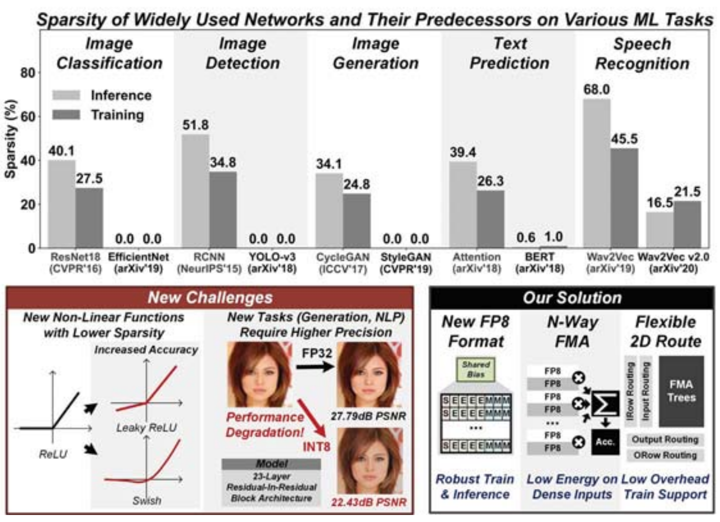


Figure 9.3.1: Overview of new challenges in mobile neural-network processors and our solution.

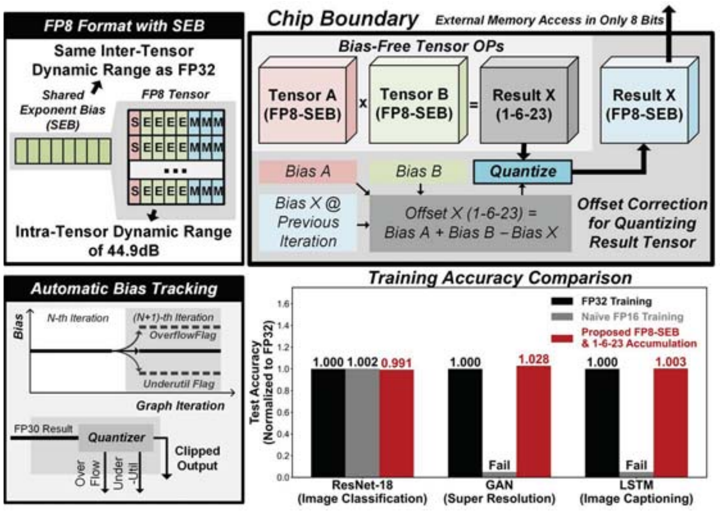

FP8 Format with SEB
Same Inter-Tensor Dynamic Range as FP32
Shared Exponent Bias (SEB)
FP8 Tensor
Intra-Tensor Dynamic Range of 44.9dB
Chip Boundary
External Memory Access in Only 8 Bits
Bias-Free Tensor OPs
Tensor A (FP8-SEB)
Tensor B (FP8-SEB)
Result X (1-6-23)
Result X (FP8-SEB)
Bias A
Bias B
Quantize
Bias X @ Previous Iteration
Offset X (1-6-23) = Bias A + Bias B − Bias X
Offset Correction for Quantizing Result Tensor
Automatic Bias Tracking
N-th Iteration
(N+1)-th Iteration
OverflowFlag
Underutil Flag
Graph Iteration
FP30 Result
Quantizer
Clipped Output
Training Accuracy Comparison
FP32 Training
Naive FP16 Training
Proposed FP8-SEB & 1-6-23 Accumulation
Test Accuracy (Normalized to FP32)
Fail
ResNet-18 (Image Classification)
GAN (Super Resolution)
LSTM (Image Captioning)

Figure 9.3.2: The proposed training scheme using FP8-SEB with FP30 accumulation matches FP32 training performance.

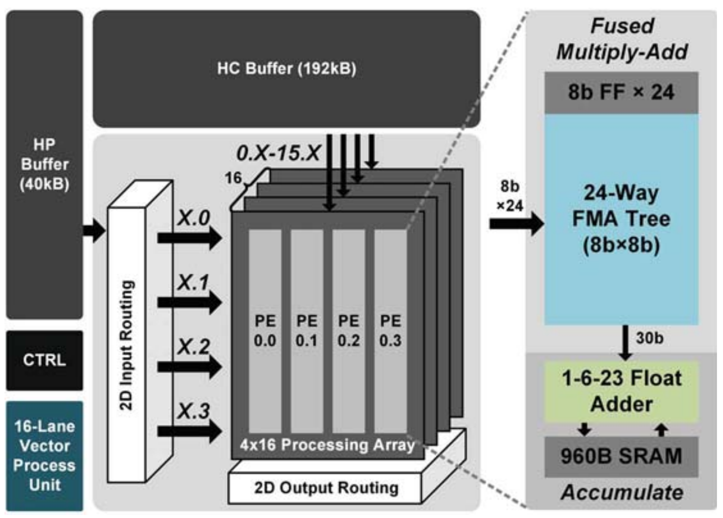


Figure 9.3.3: Architecture of the training processor.

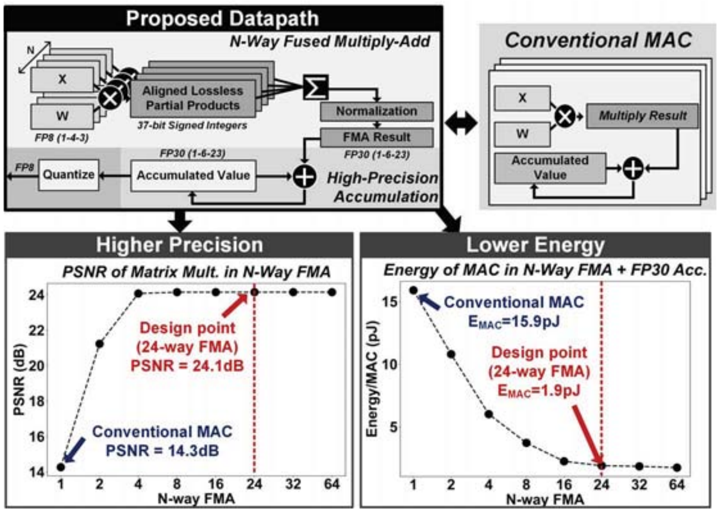

Proposed Datapath
N-Way Fused Multiply-Add
Aligned Lossless Partial Products
37-bit Signed Integers
Normalization
FMA Result
FP30 (1-6-23)
Quantize
Accumulated Value
High-Precision Accumulation
Conventional MAC
Multiply Result
Higher Precision
PSNR of Matrix Mult. in N-Way FMA
Design point (24-way FMA) PSNR = 24.1dB
Conventional MAC PSNR = 14.3dB
N-way FMA
Lower Energy
Energy of MAC in N-Way FMA + FP30 Acc.
Conventional MAC E_MAC=15.9pJ
Design point (24-way FMA) E_MAC=1.9pJ
Energy/MAC (pJ)

Figure 9.3.4: Efficient implementation of processing element using N-way Fused Multiply-Add (FMA) trees.

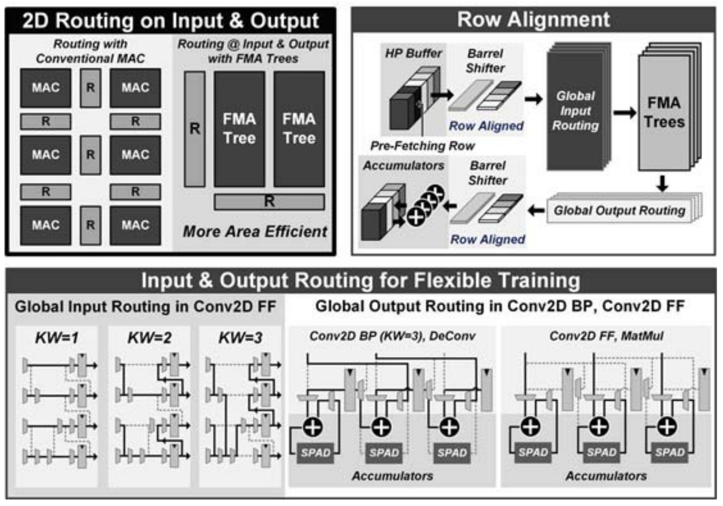
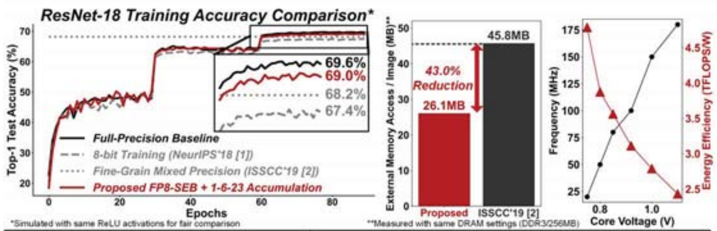


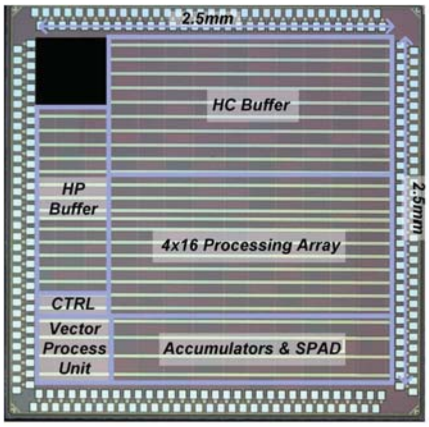
Figure 9.3.5: 2D routing on input and output points for flexible training on CNN, FC and RNN.



	This Work	ISSCC'19 [2]	ISSCC'20 [3]	ISSCC'19 [4]	S.VLSI'18 [5]	S.VLSI'20 [6]
Technology	**40nm**	65nm	65nm	65nm	14nm	14nm
Training	**CNN/FC/RNN**	CNN/FC/RNN	DNN(GAN)	CNN/FC/RNN	CNN/FC/RNN	CNN/FC/RNN
Data Format	**FP8-SEB**	FP8/FP16	FP8/FP16	Bfloat16	Ternary/FP16	FP16/FP32
Peak Performance***	**567GFLOPS @180MHz, 1.1V**	300-600GFLOPS	540-1080GFLOPS	204GFLOPS	1500GFOPS	3000GFLOPS
Max Energy Efficiency***	**4.81TFLOPS/W @20MHz, 0.75V**	1.74-3.48 TFLOPS/W	1.81-3.62 TFLOPS/W	2.16TFLOPS/W	N/A	1.4TFLOPS/W
Dense Model Training Efficiency***	**1.64TFLOPS/W (ResNet-18)**	0.66 TFLOPS/W (ResNet-18)	0.57 TFLOPS/W (CycleGAN)	N/A	N/A	N/A
On-Chip Memory	**293kB**	372kB	676kB	448kB	2MB	2MB
Core Area	**6.25mm^2**	16mm^2	32.4mm^2	16mm^2	9mm^2	9.84mm^2
Power	**13.1-230mW**	43-367mW	58-647mW	2.4-196mW	N/A	58-647mW

***For non-sparse activations (sparsity=0%)

Figure 9.3.6: Performance summary and comparison with prior works.



Technology	40nm LP
Die Size	2.7×2.7mm^2
Core Area	2.5×2.5mm^2
On-Chip Memory	293kB
Supply Voltage	0.75-1.1V
Frequency	20MHz @ 0.75V - 180MHz @ 1.1V
Peak Performance	567GFLOPS
Energy Efficiency	4.81TFLOPS/W
Data Format	FP8-SEB
Target Application	Mobile Inference/Training on CNN/FC/RNN

Figure 9.3.7: Die photo and summary table.