A Scalable Multi-TeraOPS Deep Learning Processor Core for AI Training and Inference

Bruce Fleischer, Sunil Shukla, Matthew Ziegler, Joel Silberman, Jinwook Oh, Vijayalakshmi Srinivasan, Jungwook Choi, Silvia Mueller[2], Ankur Agrawal, Tina Babinsky[2], Nianzheng Cao, Chia-Yu Chen, Pierce Chuang, Thomas Fox, George Gristede, Michael Guillorn, Howard Haynie[1], Michael Klaiber[2], Dongsoo Lee, Shih-Hsien Lo, Gary Maier[3], Michael Scheuermann, Swagath Venkataramani, Christos Vezyrtzis, Naigang Wang, Fanchieh Yee, Ching Zhou, Pong-Fei Lu, Brian Curran[1], Leland Chang, Kailash Gopalakrishnan

IBM TJ Watson Research Center, Yorktown Heights, NY; IBM Systems Group, [1]Poughkeepsie, NY; [2]Boeblingen Germany; [3]East Fishkill, NY

Abstract – A multi-TOPS AI core is presented for acceleration of deep learning training and inference in systems from edge devices to data centers. With a programmable architecture and custom ISA, this engine achieves >90% sustained utilization across the range of neural network topologies by employing a dataflow architecture and an on-chip scratchpad hierarchy. Compute precision is optimized at 16b floating point (fp16) for high model accuracy in training and inference as well as 1b/2b (binary/ternary) integer for aggressive inference performance. At 1.5 GHz, the AI core prototype achieves 1.5 TFLOPS fp16, 12 TOPS ternary, or 24 TOPS binary peak performance in 14nm CMOS.

Introduction

Driven by recent advancements in deep learning and an explosion in the use of machine learning algorithms across application domains, AI acceleration is now critically important to hardware systems from mobile/IoT devices to enterprise servers. Such AI accelerators must be sufficiently general and programmable to provide high sustained performance across neural network topologies, model sizes, and mini-batch sizes. With efficient compute engines that carefully leverage reduced precision and a parallel architecture that ensures very high utilization, we demonstrate a multi-TOPS accelerator core building block that can be scaled across a broad range of AI hardware systems.

Model accuracy is the fundamental measure of deep learning quality. Reduction of hardware numerical precision significantly reduces the area and power of compute engines, but degrades accuracy if taken too far. Precision limits are less stringent for inference than for training, but evolving AI applications such as on-device personalization or incremental learning may unify requirements. In this work, an efficient baseline (fp16) is chosen for both training and inference, but additional precisions are included to perform limited functions for model accuracy (fp32) and to aggressively boost inference performance (binary/ternary). This enables competitive power efficiency (TOPS/W) as is often reported in literature, but, just as importantly, also high raw performance (TOPS) and area efficiency (TOPS/mm^2).

A large number of efficient compute engines must be supported by sufficient bandwidth and memory structures that stream input/output data and maintain high compute utilization. This work uses a dataflow architecture, which is particularly effective for deep learning workloads, and an on-chip scratchpad hierarchy to achieve >90% sustained utilization across a broad range of neural network topologies.

Architecture Overview & Advantages

The accelerator core consists of a compute array arranged as a 2-D torus with nearest-neighbor FIFO links and a 2-level scratchpad memory (Fig. 1). Most of the nodes are fp16 processing elements (PEs), but there is an added row of fp32 special-function units (SFUs). Each compute unit (Fig. 2) contains a 16-deep local register file (LRF) and has area and power dominated by compute logic and data flip-flops. An 8 KB level-0 (L0) scratchpad is inserted in both torus dimensions. Data enters and leaves the torus through connections between the L0s and a 2 MB next-level (Lx) scratchpad, which arbitrates between local access and core I/O for scalability to multi-core SoCs. The peak read+write bandwidths of each L0 and the Lx is 192+192 GB/s. The core can execute a variety of data flows through appropriate program structure and instruction operands (LRF or FIFOs). This flexibility balances re-use of data at each level and moving data between levels to supply data to the PEs to sustain >90% utilization.

All units have carefully curated instruction sets (Table 1) and provide hardware support of nested loops. The PE ISA has a variety of fp16, ternary, binary, and bitwise operations. For the few operations needing high precision, the SFU offers an fp32 version of the PE ISA plus instructions to convert between fp16 and fp32 with a peak fp32 performance of 96 GFLOPS. Non-linear functions, which represent <5% of the total execution time in our benchmarks, can be calculated in both fp16 or fp32, with ReLU execution and estimate instructions to accelerate reciprocal, sqrt, exp, and ln.

Fig. 3 shows that the training accuracy of two state-of-the-art neural networks, ResNet-18 and ResNet-50, shows zero degradation when using fp16 instead of fp32 for most computations. For inference, weight resolution can be lowered from fp32 to ternary, which reduces required Lx storage and bandwidth by 16x while degrading accuracy by only 3% for ResNet-18 (vs. state-of-the-art [1]).

Fig. 4 shows PE utilization in the core for forward, backward, and update phases of ResNet-18 and ResNet-50 training (inference utilization is the same as forward). Note that utilization degrades gracefully as the mini-batch size decreases from 8 to 1 and is >90% for inference even with mini-batch size=1. Multiple core features contribute to such high utilization. The 2-D torus and memory hierarchy allow programs to use different dataflows such as weight-, output-, or row-stationary [2]. For example, the forward pass dataflow is row-stationary for the first convolution layer of ResNet-18 (output rows held in LRF) and weight-stationary for all other convolution layers. The Lx capacity and bandwidth allow double-buffering of blocks of inputs, weights and outputs to avoid latency penalty for these transfers. Since the application run time is dominated by convolutions, we exploit the architecture and ISA to implement them as native convolutions to avoid matrix storage and bandwidth needed for lowered GEMM implementations.

Design, Test, & Hardware Measurements

A synthesis and automated place and route tool flow was used to tape out a 14nm test chip (Fig. 5). The overall chip consists of an instance of the accelerator core integrated with infrastructure (Fig. 6) supporting through-the-pins scan and ATPG test, JTAG-controlled at-speed memory built-in self-test and clock frequency divider, and tester-driven functional operation via a chip-management unit (CMU). The chip is flip-chip C4 bonded to a laminate and tested in module form (Fig. 7). Clock dividers allow external monitoring of the received high-frequency and frequency-scaled core clock (Fig. 8).

An at-speed functional verification sequence consists of setting the functional clock to the target frequency, loading a program and data into Lx via the CMU, signaling the core to fetch and execute the loaded program, and, after successfully polling the core for program completion, offloading computed values for comparison to expectation. Functional tests cover all the operations encountered in forward and backward computation and update phases for training and inference of deep learning networks, including convolution, matrix multiplication, activation functions such as ReLU, tanh and sigmoid, and maximum- and average-pooling. Fig. 9 shows a shmoo plot summarizing all workloads and demonstrates operation between 0.58V and 0.9V spanning a frequency range from 750MHz to 1.5GHz. Measured relative power vs. FPU utilization extrapolated to zero (Fig. 10) indicates that use of the Lx and L0 to supply data every cycle accounts for only 18% of the accelerator power when computing at full utilization.

Summary

A deep learning accelerator core providing high compute efficiency and utilization is demonstrated to achieve performance and area efficiency advantages over recently published high-performance AI chips (Table 2). This core can be a practical building block for SoCs to enable a broad range of AI hardware systems.

978-1-5386-4214-6/18/$31.00 ©2018 IEEE

Acknowledgements

The authors are grateful for characterization support from J. You, D. Mundhenk, and J. Jones, design and build support from C. Baks, R. Oldrey, P. Gaschke, T. Dickson, D. Turnbull, and R. Heath, technology support from K. Lewis and G. Tellez, and executive support from J. Burns, R. Fischer, R. Puri, T.-C. Chen, D. Gil, M. Rosenfield, and M. Khare.

References

[1] C. Zhu, et al., *ICLR*, 2017.
[2] Y. H. Chen, et al., *ISSCC*, 2016.
[3] Ando, et al., *VLSI Symp.*, 2017.
[4] S. Yin, et al., *VLSI Symp.*, 2017.
[5] P. Knag, et al., *VLSI Symp*, 2016.
[6] G. Desoli, et al., *ISSCC*, 2017.
[7] N. P. Jouppi, et al., *ISCA*, 2017.

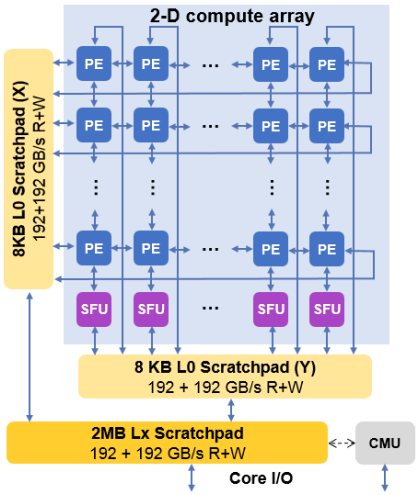

Figure 1: Accelerator core compute array

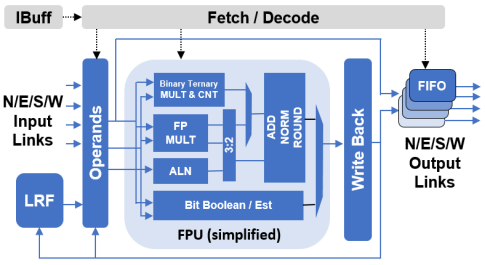

Figure 2: PE/SFU block diagram

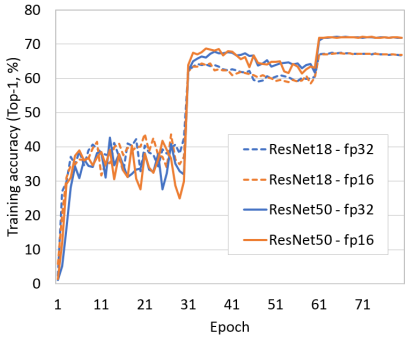

Figure 3: 16b training accuracy

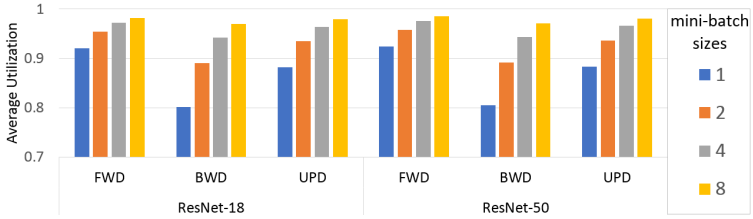

Figure 4: Core utilization: ResNet-50 and ResNet-18

Category	PE	SFU
FP multiply-add/sub, min/max, compare, select, convert <-> int	Y	Y
FP estimate for recip/sqrt/exp/ln	Y	Y
Binary/Ternary multiply-sum-acc.	Y	Y
Bitwise boolean	Y	Y
FP32 <-> FP16 conversions		Y

Table 1: Instruction categories

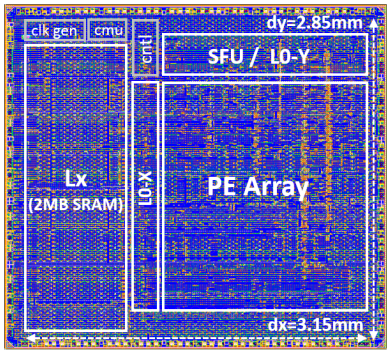

Figure 5: Test chip floorplan and dimensions

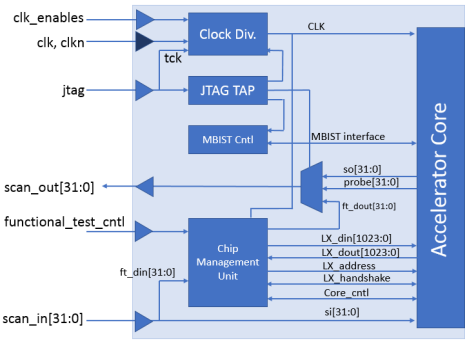

Figure 6: Test infrastructure block diagram

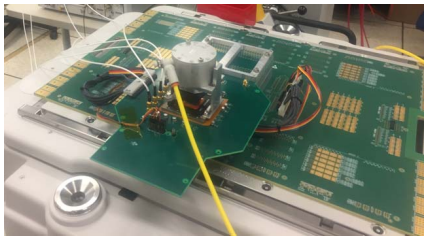

Figure 7: Module on the tester

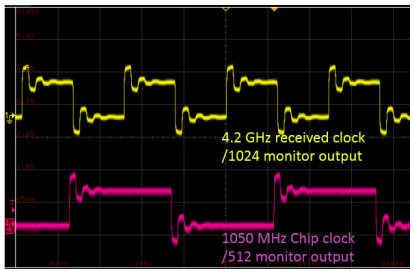

Figure 8: Externally monitored clock signals

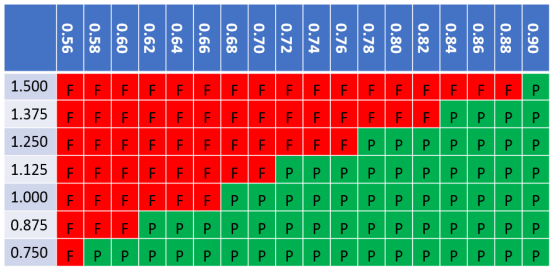

Figure 9: Shmoo plot

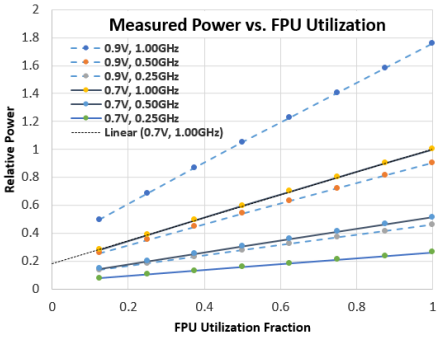

Figure 10: Relative power vs. FPU utilization

	This Work		[2]	[3]	[4]	[5]	[6]	[7]
Technology	14nm		65nm	65nm	65nm	40nm	28nm FDSOI	28nm
Evaluation Application	CNN/MLP/LSTM		AlexNet	handwritten digit recognition	AlexNet	facial recognition	AlexNet	CNN/MLP /LSTM
Supply Voltage (V)	0.58	0.9	1	0.55-1.0	1.2	0.65 0.9	0.575-1.1	unknown
Frequncy (GHz)	0.75	1.5	0.2	0.1-0.4	0.2	0.12 0.24	0.2-1.175	0.7
Area (mm^2)	9		12.3	3.9	19.36	1.4	34	<=331
On-chip Memory (MB)	2		0.182	unknown	0.348	unknown	>4	28 (MiB)
Precision Training	fp16		N/A	N/A	N/A	N/A	N/A	N/A
Precision Inference	Binary / Ternary / fp16		16b fixed	Binary / Ternary	8b, 16b, or <8b	8b mult, 16b add	CA = 16b fixed DSP = 32b	8b integer
Training Perf. (TFLOPS)	0.75	1.5	N/A	N/A	N/A	N/A	N/A	N/A
Inference Perf. (TOPS)	12 / 6 / 0.75	24 / 12 / 1.5	0.07	1.38	0.368	0.449 0.898	0.676	92
FLOPS / mm^2 (Training)	0.08	0.17	N/A	N/A	N/A	N/A	N/A	N/A
TOPS / mm^2 (Inference)	1.33 / 0.67 / 0.08	2.67 / 1.33 / 0.17	0.006	0.365	0.019	0.321 0.642	0.020	>=0.278

Table 2: Comparison of this work to recently published cores [2-5] and SoCs [6-7]