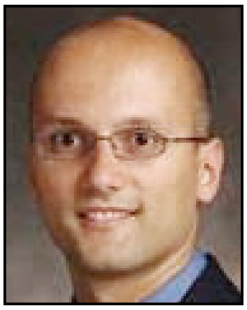
Session 13 Overview: *Machine Learning and Signal Processing*

DIGITAL ARCHITECTURES AND SYSTEMS SUBCOMMITTEE



Session Chair:
Dejan Marković
University of California, Los Angeles, Los Angeles, CA

Associate Chair:
Masato Motomura
Hokkaido University, Sapporo, Japan

Subcommittee Chair: *Byeong-Gyu Nam,* *Chungnam National University, Daejeon, Korea*

Architectures supporting machine learning for embedded perception and cognition are continuing their rapid evolution, inspired by modern data analytics and enabled by the low energy cost of CMOS processing. This makes it feasible to migrate data analytics toward edge and wearable devices. To further support increased requirements for multiuser connectivity and sparse data, multi-user MIMO and compressive reconstruction are also required.

This session covers trends in machine learning and signal processing for improved accuracy of speech, image, video processing for next-generation mobile/edge and data center devices. The session features programmable accelerators for Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Multi-Layer Perceptron (MLP) algorithms, with digital and mixed-signal processing kernels. The session concludes with a link-adaptive massive MIMO detector and robust compressive-sensing reconstruction processors.

INVITED PAPER

1:30 PM

13.1 A Shift Towards Edge Machine-Learning Processing

O. Temam, Google, Paris, France

The field of machine learning, especially Deep Neural Networks, is advancing at a breathtaking pace, with new functionalities achieved on a monthly basis. In the span of a few years, close to human-level accuracy has been achieved for simple voice commands, then full speech recognition, speech synthesis, translation, and increasing progress has been achieved in language understanding.

Machine-learning researchers largely acknowledge that the current successes of deep neural networks have been fueled by two evolutions: the availability of a large quantity of data for training, and the availability of high-performance computing at low cost, initially enabled by GPUs. Both advances combined to make training times tractable for large neural-network models.

Beyond GPUs, both the broad application span and high computational cost of deep neural networks have made custom machine-learning hardware economically sensible, and such architectures are currently being developed by many hardware, cloud or startup companies. The number of competing companies and the broad dissemination of knowledge on how to design such hardware should help reduce the cost of neural network computing, and make sophisticated machine learning more accessible in the coming years.

As Moore's Law plateaus, one of the main paths forward is increasing customization for increased efficiency. This trend will paradoxically first arise at the edge (vs. in the data center), where hardware efficiency is most critical. Unfortunately, the need for customization/efficiency runs contrary to the very fast evolution of machine-learning algorithms. Traditional architectural approaches for achieving generality, while great for general-purpose computing, may not be best suited for resolving this tension between efficiency, velocity and generality.

Beyond hardware efficiency challenges, the other key challenge remains access to data. Consumer data privacy, corporate data confidentiality, or even regulatory compliance force a shift towards processing data closer to where it exists, i.e., at the edge. For many applications, doing so also provides useful, if not indispensable, latency, bandwidth and connectivity benefits for many applications. It even provides an out-of-the-box way to tackle the economic consequences of a plateauing Moore's Law for fast growing data-center machine-learning applications.

The talk will go over these different trends and their consequences.

2:00 PM

13.2 QUEST: A 7.49TOPS Multi-Purpose Log-Quantized DNN Inference Engine Stacked on 96MB 3D SRAM Using Inductive-Coupling Technology in 40nm CMOS

K. Ueyoshi, Hokkaido University, Sapporo, Japan

In Paper 13.2, Hokkaido University presents a 14.3×8.5mm² multi-purpose log-quantized deep neural network (DNN) inference engine stacked on a 96MB 3D SRAM using inductive coupling technology in 40nm. The system features 3-cycle 28.8GB/s memory communication and 7.49TOPS peak performance in binary precision at 1.1V, 300MHz, for cutting-edge DNN workloads.

2:30 PM

13.3 UNPU: A 50.6TOPS/W Unified Deep Neural Network Accelerator with 1b-to-16b Fully-Variable Weight Bit-Precision

J. Lee, KAIST, Daejeon, Korea

In Paper 13.3, KAIST describes a DNN accelerator with variable bit precision from 1b to 16b. Using a flexible DNN core architecture, look-up-table-based bit-serial processing, and off-chip memory management, the 16mm² 65nm chip achieves 50.6TOPS/W energy efficiency for 1b data at 10MHz, 0.66V.

3:15 PM

13.4 A 9.02mW CNN-Stereo-Based Real-Time 3D Hand-Gesture Recognition Processor for Smart Mobile Devices

S. Choi, KAIST, Daejeon, Korea

In Paper 13.4, KAIST presents a 3D hand-gesture recognition processor for real-time user interaction in smart mobile devices. With a CNN stereo engine, triple ping-pong buffers, and processor-in-memory techniques, the 16mm² 65nm processor achieves real-time 3D hand-gesture recognition with 9.02mW and 4.3mm error at 0.85V, 50MHz.

3:45 PM

13.5 An Always-On 3.8μJ/86% CIFAR-10 Mixed-Signal Binary CNN Processor with All Memory on Chip in 28nm CMOS

D. Bankman, Stanford University, Stanford, CA

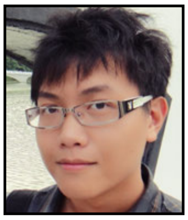
In Paper 13.5, Stanford University and KU Leuven introduce a mixed-signal binary CNN processor based on near-memory computing. The 2.4×2.4mm² 28nm 0.6V processor features 328KB of on-chip SRAM for a 9-layer CNN, data parallelism and parameter re-use, achieving a 3.8μJ/classification at 86.05% accuracy on the CIFAR-10 dataset.

4:15 PM

13.6 A 1.8Gb/s 70.6pJ/b 128×16 Link-Adaptive Near-Optimal Massive MIMO Detector in 28nm UTBB-FDSOI

W. Tang, University of Michigan, Ann Arbor, MI

In Paper 13.6, the University of Michigan describes 128×16 massive MIMO detector with link adaptation to meet practical channel conditions with scalable energy. Implemented as a condensed systolic array, the 2mm² 28nm FDSOI chip achieves 1.8Gb/s at 70pJ/b, 569MHz and 4.3dB processing gain with channel data obtained from real-life measurements.



4:45 PM

13.7 A 232-to-1996KS/s Robust Compressive-Sensing Reconstruction Engine for Real-Time Physiological Signals Monitoring

T-S. Chen, National Taiwan University, Taipei, Taiwan

In Paper 13.7, National Taiwan University presents a compressive-sensing reconstruction engine with parallel atom searching approach to reduce signal distortion due to measurement noise. The 2.93×2.93mm² 40nm processor achieves up to 1996KS/s with 93mW power consumption at 0.9V, 67.5MHz.

13.2 QUEST: A 7.49TOPS Multi-Purpose Log-Quantized DNN Inference Engine Stacked on 96MB 3D SRAM Using Inductive-Coupling Technology in 40nm CMOS

Kodai Ueyoshi[1], Kota Ando[1], Kazutoshi Hirose[1], Shinya Takamaeda-Yamazaki[1], Junichiro Kadomoto[2], Tomoki Miyata[2], Mototsugu Hamada[2], Tadahiro Kuroda[2], Masato Motomura[1]

[1]Hokkaido University, Sapporo, Japan
[2]Keio University, Yokohama, Japan

A key consideration for deep neural network (DNN) inference accelerators is the need for large and high-bandwidth external memories. Although an architectural concept for stacking a DNN accelerator with DRAMs has been proposed previously, long DRAM latency remains problematic and limits the performance [1]. Recent algorithm-level optimizations, such as network pruning and compression, have shown success in reducing the DNN memory size [2]; however, since networks become irregular and sparse, they induce an additional need for agile random accesses to the memory systems.

Figure 13.2.1 illustrates our proposal: stacking a DNN inference engine, QUEST, with multi-vault SRAMs using inductive-coupling die-to-die wireless communication technology, known as a ThruChip Interface (TCI) [3]. Parallel TCI channels placed in a planar manner provide QUEST with multiple independent high-bandwidth access points to the stacked SRAMs. SRAMs can provide random access capability with extremely low latency (an order of magnitude lower than DRAMs), whereas 3D stacking helps SRAMs achieve reasonably large memory capacity. In Fig. 13.2.1, QUEST and 8 SRAMs are TCI-stacked as a single 14.3×8.5 mm^2 3D module. Power/ground are supplied through TSVs. QUEST has 24 processing cores running at 300MHz, where each core is associated with one 32b-width 4MB SRAM vault. Running at 3.6GHz, a TCI channel (7-Tx/5-Rx coils) realizes 9.6Gb/s/vault, combined 28.8GB/s/module, R/W data bandwidth in a source synchronous manner. The R/W latency including TCI trip time is 3 cycles, which is uniform over the 8 SRAMs. TSV technology, used commonly for die stacking, is known to experience open-contact failure. In our design, however, since all signal transmissions are conducted by wireless TCI channels, the presented 3D module can limit the usage of TSVs to power/ground grids, where numerous parallel connections negate this concern.

Figure 13.2.2 shows the overall block diagram of the fabricated QUEST prototype. The 24 cores run in a MIMD-parallel manner, where inter-core communication is handled either with a mesh-structured local link or tree-structured global network. Each core has a micro-programmed sequencer for setting and controlling the PE array. Synchronization among the cores is managed through a synchronization table when needed. Each core also has a DMAC, which issues memory accesses to intra-core memories (shaded) and the stacked SRAM vault in response to intra/inter-core memory requests. The 32×16 PE array features a bit-serial architecture: the PE conducts binary computation in a single cycle, and N-bit log-quantized ones in N cycles (N<5). Weights double-buffered in W_MEMs are distributed to the PE array in a fully parallel manner, whereas incoming activations also double-buffered in A_MEMs are broadcast in a row-parallel manner. The ACT unit at the tail of each column applies the bias (shifted-in from B_MEMs), scaling, and activation function; and then writes the output activations into O_MEMs.

Unlike other array-structured DNN accelerators, all PEs receive unique weight bits, whereas the PEs in a row receive an identical activation bit, as detailed in Fig. 13.2.3. In a PE column, partial dot products are first generated in parallel in PEs and then simply shifted towards ACT where they are accumulated. The pipelined shifts hide behind the PE-parallel, bit-serial, dot-product computations as shown in the time chart. These mechanisms are key enablers for handling various DNNs on a single homogeneous PE array (whereas [5] and [6] use hybrid cores and hybrid PEs, respectively, for different DNN types): *e.g.,* for a fully connected (FC) layer in a CNN, MLP, or RNN, up to 32 fan-ins for a neuron are mapped onto a PE column at a time and then time multiplexed on the same column. For a convolutional (CONV) layer in CNN, on the other hand, up to 32 input channels are mapped onto a PE column at a time and then time multiplexed. The filter kernel is stored vertically in the W_MEMs and processed in an element-by-element, kernel-parallel manner.

Figure 13.2.4 presents a log-quantized neural datapath. The log-quantization method [4] is superior to linear quantization in two ways: 1) its "denser the finer" approach allows it to represent weight/activation distributions better, and 2) resource-consuming multiply operations are reduced to additions. Dot-products are computed in PEs by "log" bit-serial addition and "linear" accumulation. ACT accumulates the dot products and adds a bias in "linear", then applies a scaling/activation function such as ReLU in "log". The lightweight PE architecture has enabled the dense PE array to be tightly coupled (bit-by-bit) with W_MEMs and A_MEMs (Fig. 13.2.2), achieving versatile parallel NN computation (Fig. 13.2.3). Log-quantization inference accuracy is evaluated on AlexNet (for ImageNet) and on LeNet-5 (for MNIST). Log-4 (4b log-quantized) AlexNet shows only marginal degradation compared with FP-32, whereas Log-3 was destructive. Even binary can attain reasonable accuracy for LeNet-5: the performance-accuracy trade-off is also indicated in the figure.

Figure 13.2.5 depicts AlexNet mapped on the QUEST 3D module, where 24 parallel cores process the inference in a layer-by-layer manner, producing/reading intermediate results to/from the SRAMs, respectively. For a CONV layer, an output channel is mapped spatially among the cores so that a "cluster" of cores can share same input channels. For a FC layer, output neurons are mapped onto all the cores evenly, requiring all-to-all shuffling data-distribution patterns. In both cases, computation in a core must read activations from another core's SRAM vaults, that are delivered through the TCI channels and the on-chip networks. The accesses are scattered across individual memory spaces, and burst lengths are very short (1 to 4 for this mapping). Fig. 13.2.5 summarizes the performance of the Log-4 AlexNet, which occupies 39% of the 3D SRAM, as well as Log-4 and binary VGG11 (for CIFAR-10). It is shown that for AlexNet, having more than 2.9MB of on-chip memory is crucial for sustaining above 90% effective/peak performance. The 3-cycle short random-access latency of the 3D SRAM, on the other hand, is also indispensable for effective performance, assuming burst memory access with 30-cycle initial latency, which mimics modern DRAM latency, effective VGG11 performance degrades drastically for Log-4 and binary cases. Larger DNNs such as ResNet, moreover, require aggressive pruning to fit the limited memory space, where the presented random-access capability of the 3D module will become even more indispensable.

Figure 13.2.6 compares recently reported multi-purpose (CNN/FC/RNN, etc.) DNN accelerators [5] and [6], using LUT-based and linear quantization, respectively, with this work. Those works integrated a limited amount of on-chip SRAM (around 300KB), and did not include external memory for power estimation. QUEST, on the other hand, integrates 7.68MB large on-chip SRAM (sufficient for AlexNet on-chip buffering) in addition to the 96MB 3D SRAM. It achieves 5× better effective performance on AlexNet benchmark at 4b precision. Since external memory accesses are responsible for the majority of the power dissipation, and since the 3D SRAM can substantially reduce external memory power in comparison to DRAMs, system-level energy efficiency favors the proposed solution.

Figure 13.2.7 shows a QUEST prototype microphotograph with a specification table. To summarize, QUEST is aimed toward rapidly revolutionizing highly compressed (bit-reduced, pruned, etc.) DNNs with three main architectural features: 1) 3D integration with large capacity/bandwidth yet low-latency random access SRAM, 2) flexible dataflow support in the PE array for CONV/FC and other types of DNN layers, 3) a bit-serial PE architecture for binarized and log-quantized DNN representations.

Acknowledgements:
This work was funded by JST ACCEL Grant Number JPMJAC1502, Japan. The authors thank Profs. T. Asai, M. Ikebe, E. Sano, M. Arita from Hokkaido University and the colleagues at UltraMemory Inc. for their invaluable support.

References:
[1] M. Gao, et al., "TETRIS: Scalable and Efficient Neural Network Acceleration with 3D Memory," *ACM ASPLOS*, pp. 751-764, 2017.
[2] A. Parashar, et al., "SCNN: An Accelerator for Compressed-sparse Convolutional Neural Networks," *ACM ISCA*, pp. 27-40, 2017.
[3] D. Ditzel, et al., "Low-Cost 3D Chip Stacking with ThruChip Wireless Connections," *IEEE Hot Chips*, pp. 1-37, 2014.
[4] D. Miyashita, et al., "Convolutional Neural Networks using Logarithmic Data Representation," *arXiv: 1603.01025 [cs.NE]*, 2016.
[5] D. Shin, et al., "DNPU: An 8.1TOPS/W Reconfigurable CNN-RNN Processor for General-Purpose Deep Neural Networks," *ISSCC*, pp. 240-241, 2017.
[6] S. Yin, et al., "A 1.06-to-5.09 TOPS/W Reconfigurable Hybrid-Neural-Network Processor for Deep Learning Applications," *IEEE Symp. on VLSI Circuits*, 2017.

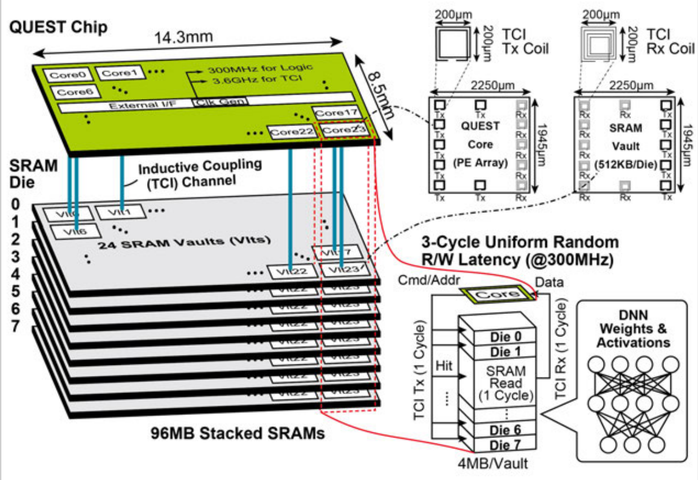


Figure 13.2.1: QUEST module overview and 3-cycle R/W latency on 3D SRAM.

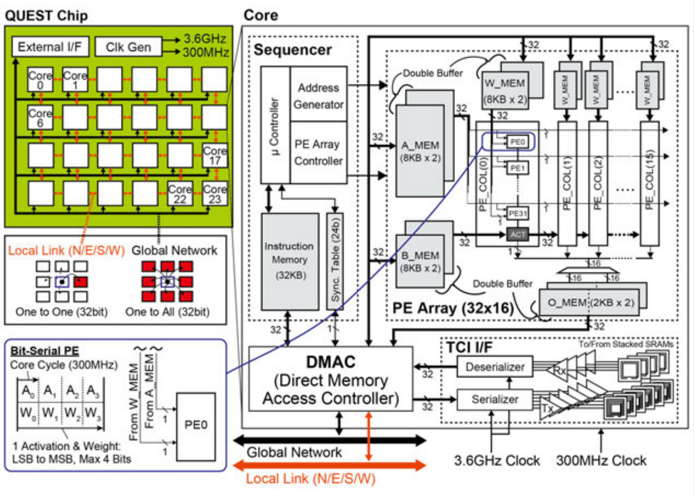


Figure 13.2.2: Overall architecture of the proposed DNN inference engine (QUEST).

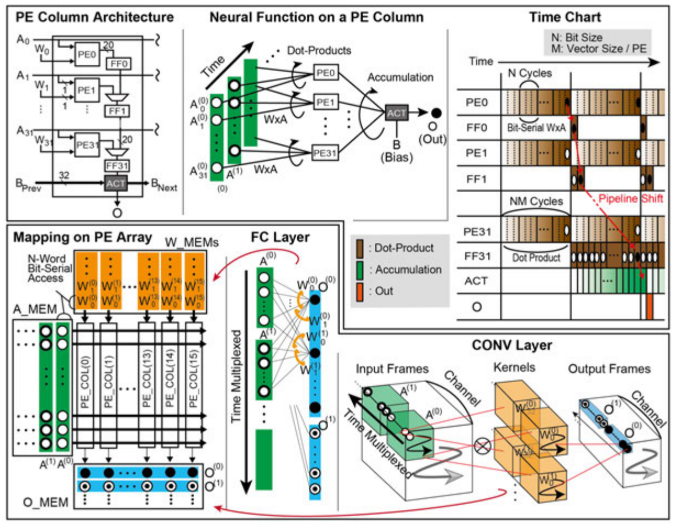


Figure 13.2.3: Detailed computation in a PE column (top). FC/CONV dataflow on a homogeneous PE array (bottom).

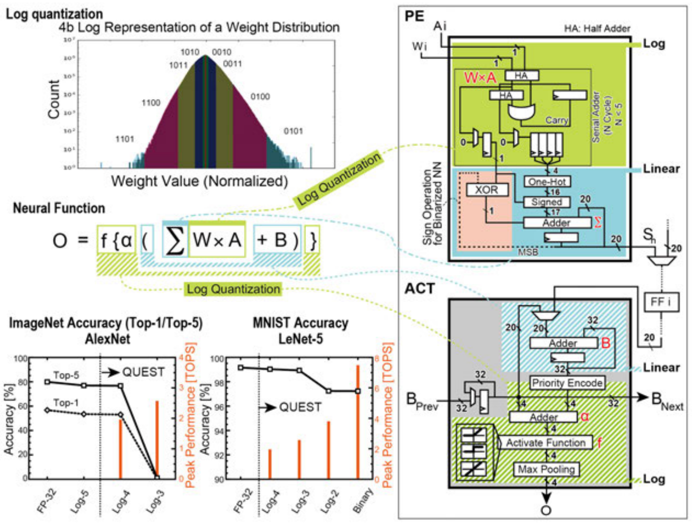


Figure 13.2.4: A log-quantized neural datapath in PE and ACT, and accuracy evaluation on realistic networks.

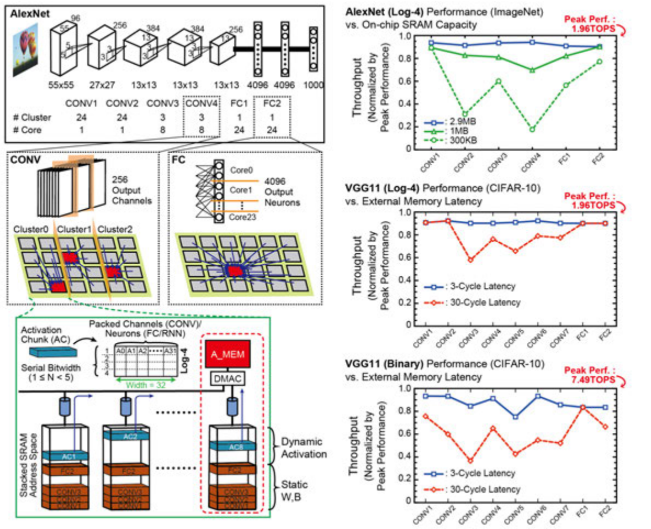
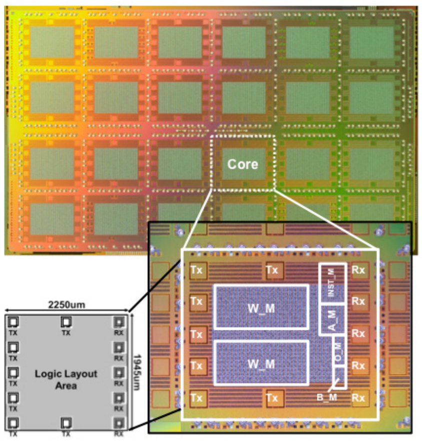


Figure 13.2.5: AlexNet mapped on the QUEST (left). Throughput across different memory capacity/latency (right).

	ISSCC2017 [5]		VLSIC2017 [6]	This Work
Technology	CMOS 65nm		CMOS 65nm LP	CMOS 40nm LP
Die Area [mm²]	4 x 4		4.4 x 4.4	14.3 x 8.5
Target DNN	CNN/MLP/RNN		CNN/MLP/RNN	CNN/MLP/RNN
Precision [bit]	4 - 16		2 - 16	1 - 4
Quantization	LUT-based		Linear	Logarithmic
Clock Freq. [MHz]	50 - 200		100 - 400	75 – 330
Supply Voltage [V]	0.77 - 1.1		0.67 - 1.2	0.77 – 1.1
External Memory	Not Discussed (Not Included in Power Dissipation)			3D-Stacked SRAM
Power Dissipation [W]	0.03@0.77V 0.28@1.1V		0.04@0.67V 0.45@1.2V	3.3@1.1V
On-chip SRAM [KB]	290		349	7,680
Peak Performance [TOPS]	0.3@16b 1.2@4b	CONV	0.410@4b	1.96@4b 7.49@1b
	0.025@4b	FC		
AlexNet@4b Effective Performance [TOPS]	0.26	CONV	0.368	1.825
	0.02	FC		
AlexNet @4b Top-5 Accuracy	Not Shown			76.7% (-3.2% from FP-32)
VGG11 Effective Perf. [TOPS]	-		-	1.78@4b 6.52@1b

Figure 13.2.6: Comparison with state-of-the-art multi-purpose DNN accelerators.



Technology	CMOS 40nm LP
Die Area	14.3 x 8.5 mm^2
Gate Count	8.15M NAND2
SRAM Capacity	7.68MB (On Chip) 96MB (In Module)
SRAM Bandwidth	604.8GB/s (On Chip) 28.8GB/s (In Module)
# of Cores	24
Core	512 PEs, 16 ACTs 289K NAND2 Gate 320KB SRAM 1 TCI Channel (7-Tx/5-Rx Coils)
Peak Performance	1.96TOPS @4b 7.49TOPS @1b [MulAdd = 2 Ops.]
Clock Freq. @1.1V	Core: 300MHz TCI: 3.6GHz
Power@1.1V	3.3W

Figure 13.2.7: A microphotograph of the QUEST prototype, along with the chip specification summary.

13.3 UNPU: A 50.6TOPS/W Unified Deep Neural Network Accelerator with 1b-to-16b Fully-Variable Weight Bit-Precision

Jinmook Lee, Changhyeon Kim, Sanghoon Kang, Dongjoo Shin, Sangyeob Kim, Hoi-Jun Yoo

KAIST, Daejeon, Korea

Deep neural network (DNN) accelerators [1-3] have been proposed to accelerate deep learning algorithms from face recognition to emotion recognition in mobile or embedded environments [3]. However, most works accelerate only the convolutional layers (CLs) or fully-connected layers (FCLs), and different DNNs, such as those containing recurrent layers (RLs) (useful for emotion recognition) have not been supported in hardware. A combined CNN-RNN accelerator [1], separately optimizing the computation-dominant CLs, and memory-dominant RLs or FCLs, was reported to increase overall performance, however, the number of processing elements (PEs) for CLs and RLs was limited by their area and consequently, performance was suboptimal in scenarios requiring only CLs or only RLs. Although the PEs for RLs can be reconfigured into PEs for CLs or vice versa, only a partial reconfiguration was possible resulting in marginal performance improvement. Moreover, previous works [1-2] supported a limited set of weight bit precisions, such as either 4b or 8b or 16b. However, lower weight bit-precisions can achieve better throughput and higher energy efficiency, and the optimal bit-precision can be varied according to different accuracy/performance requirements. Therefore, a unified DNN accelerator with fully-variable weight bit-precision is required for the energy-optimal operation of DNNs within a mobile environment.

In this paper, we present a unified neural processing unit (UNPU) supporting CLs, RLs, and FCLs with fully-variable weight bit-precision from 1b to 16b. As shown in Fig. 13.3.1, the reuse of input features (IFs) is more efficient than the reuse of weights under low-weight bit-precision and the operations of CLs become identical to those of RLs and FCLs when the IFs of the CLs are vectorized into a 1-dimensional vector so that the hardware can be fully shared in the UNPU by IF reuse. Moreover, the lookup-table-based bit-serial PE (LBPE) is implemented for energy-optimal DNN operations with variable-weight bit-precisions from 1b to 16b through iterations of 1b weight operations. Furthermore, an aligned feature loader (AFL) minimizes the amount of off-chip memory accesses required to fetch IFs by exploiting the data locality among convolution operations.

Figure 13.3.2 shows the overall architecture of the UNPU. It consists of 4 DNN cores, an aggregation core, a 1D SIMD core, and a RISC controller. All of these components are connected to an on-chip network for communication. Each DNN core has 6 LBPEs, 6 AFLs (64×6), a weight memory (48KB), an instruction decoder and a controller. The LBPE receives aligned IF as an input operand through AFLs and calculates 576 (4×12×12) multiplications in parallel in a bit-serial manner. The partial-sums (Psums) calculated by each DNN core are aggregated to an output feature (OF) in the aggregation core. The 1D SIMD core performs the remaining operations, such as non-linear activation or pooling, and the results are stored in off-chip memory through the external gateways.

Figure 13.3.3 elaborates on workload allocation. For RLs and FCLs, its 1D IF is mapped to AFLs with one-to-one (48×1) and sent to a PE. The weights are loaded from 12 channels of OF (48×12b) to calculate multiple channels of Psums with the same IF. For a CL, IFs distributed over multiple input channels are concatenated into a 1D row vector and loaded into the AFLs, as is done with RLs and FCLs. The weights of CLs are converted into 1D column vectors and then the Psums are calculated by multiplying with the 1D IF row vector. 4 LBPEs in a DNN core calculate the product between 48 pairs of IFs and weights, and each LBPE corresponds to 12 IF-weight pairs. The IF is reused for multiple column vectors from other channels. The Psums from each PE are accumulated by 12 adder trees. The weights are reused among the 6 LBPEs for better energy efficiency. For example, in RLs and FCLs, the 6 different IFs are assigned to 6 LBPEs in parallel with the same weights if batch-wise parallelism is possible. For a CL, the 6 consecutive IFs in the same channel are multiplied with the same weights in 6 different LBPEs in parallel. Peak performance for CLs and RLs (or FCLs) is increased by 1.15× and 13.8×, respectively, compared to [1] owing to the higher compute density of the unified DNN core.

Figure 13.3.4 shows the architecture of the LBPE. The key idea of the LBPE is that partial-sums are repeatedly calculated during the weight bit-serial MAC operation. A LBPE consists of 4 PE clusters, adder trees to accumulate the results of each PE cluster, and shift-and-add logic for bit-serial multiplications. Each PE cluster contains 4 look-up-table (LUT) modules and a controller that determines whether the value from LUTs is added or subtracted. In the LUT module, a table with 8 entries is used, supporting 3-way MAC for multi-bit multiplication and 4-way MAC for 1b multiplication. The LUT is updated after IFs load into the AFLs, and IF values are reused for all output channels of the layer currently being processed. The 1b weight Psums are fetched from the LUT prepared in advance and accumulated for MAC operation. The LUT can fetch 12 Psums in parallel so that a total of 48×12 Psums (64×12 for 1b case) can be calculated simultaneously on a LBPE in 1 cycle. With the help of table-based operations, the LBPE improves energy efficiency more than conventional bit-serial PEs [4]. When IFs are reused 1024 times, the energy-consumption of LBPEs, including the LUT update, is reduced by 23.1%, 27.2%, 41.0%, and 53.6%, for the case of 16b, 8b, 4b, and 1b weight operations, respectively, compared with fixed-point MAC units under the same throughput conditions.

Figure 13.3.5 explains the AFL. 6 AFL-LBPE pairs are integrated in a DNN core and each AFL has 64 entries. The data in the AFL can be shifted diagonally across AFL boundaries, as well as shifted inside the AFL itself. In the case of CLs with 3×3 kernels and stride 1, 8 entries of IF from Ch. 1 are loaded on AFL 0 at first. At the next cycle, the 7 top entries of AFL 0, except the top-most entry, are shifted diagonally to AFL 1, while the 8 entries from Ch. 2 are concatenated below the remaining 3 entries of Ch. 1 on AFL 0. And then, 8 entries from Ch. 3 are concatenated below the remaining 3 entries of Ch. 2 on AFL 0, while 6 entries on AFL 1 from Ch. 1 are shifted diagonally to AFL 2, and 7 entries from Ch. 2 are shifted diagonally to concatenate below the remaining 3 entries from Ch. 1 on AFL 1. Iterations of diagonal shifts allocate a 3×3 kernel to each AFL or LBPE so that parallel multiplication is possible to accelerate convolution. Varied stride sizes are supported via the application of multiple shifts. The AFL keeps the PE utilization high, unlike an architecture that moves data only between PEs. In addition, it can skip zeros by an upward-shift within the buffer. When the AFL is applied to AlexNet and VGG-16, external memory access operations for IF load are reduced by 57.2% and 55%, respectively.

Figure 13.3.6 shows measurement results for the fabricated UNPU. The UNPU can operate at 0.63-to-1.1V supply voltage with a maximum 200MHz clock frequency. The power consumption at 0.63V and 1.1V is 3.2mW and 297mW, respectively. The power-efficiency, as measured on CLs (5×5 kernels) with consideration of PE utilization is 3.08, 11.6, and 50.6TOPS/W for the case of 16b, 4b, and 1b weights, respectively. The architecture supports any weight bit-precision from 1b to 16b for optimal DNN operation and shows 1.43× higher power efficiency for CLs at 4b weight compared to [1]. When operating on a 1b weight network, it achieves 8.43× higher efficiency and 7.4× higher peak performance as compared to [6].

The UNPU is fabricated using 65nm CMOS technology and occupies 16mm^2 die area, as shown in the Fig. 13.3.7. The UNPU has been demonstrated successfully on facial expression recognition and dialogue generation tasks with the FER2013 and the Twitter dialogue database for human-computer interaction, respectively.

References:

[1] D. Shin, et al., "DNPU: An 8.1 TOPS/W Reconfigurable CNN-RNN Processor for General-Purpose Deep Neural Networks," *ISSCC*, pp. 240-241, 2017

[2] B. Moons, et al., "Envision: A 0.26-to-10TOPS/W Subword-Parallel Dynamic-Coltage-Accuracy-Frequency-Scalable Convolutional Neural Network processor in 28nm FDSOI," *ISSCC*, pp. 246-247, 2017.

[3] K. Bong, et al., "A 0.62mW Ultra-Low-Power Convolutional-Neural-Network Face-Recognition Processor and a CIS Integrated with Allways-On Haar-Like Face Detector," *ISSCC*, pp.248-249, 2017

[4] P. Judd, et al., "Stripes: Bit-serial Deep Neural Network Computing," *IEEE Computer Architecture Letters*, vol. 16, no. 1, pp. 80-83, Jan.-June 1 2017.

[5] S. Yin, et al., "A 1.06-to-5.09 TOPS/W Reconfigurable Hybrid-Neural-Network Processor for Deep Learning Applications," *IEEE Symp. VLSI Circuits*, 2017.

[6] K. Ando, et al., "BRein memory: A 13-Layer 4.2 K Neuron/0.8 M Synapse Binary/Ternary Reconfigurable In-Memory Deep Neural Network Accelerator in 65 nm CMOS," *IEEE Symp. VLSI Circuits*, 2017.

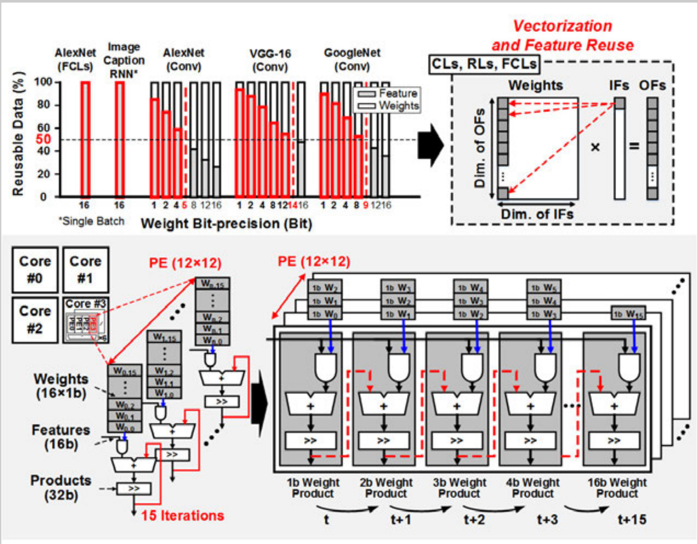


Figure 13.3.1: Fully reconfigurable unified DNN accelerator with bit-serial PEs.

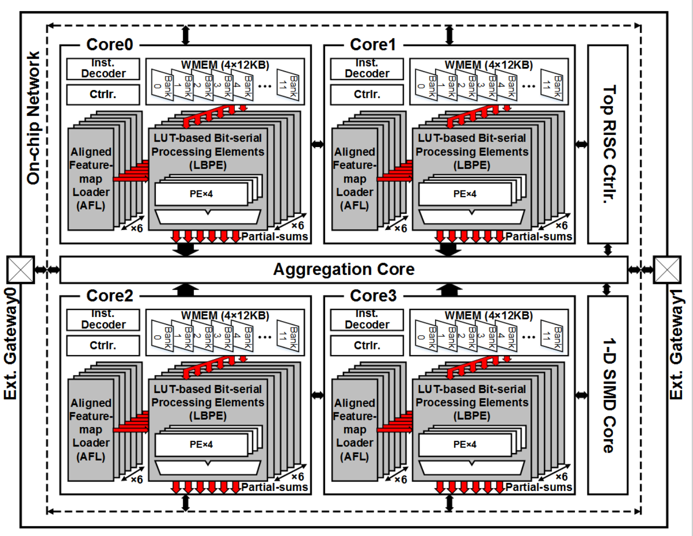


Figure 13.3.2: Overall architecture.

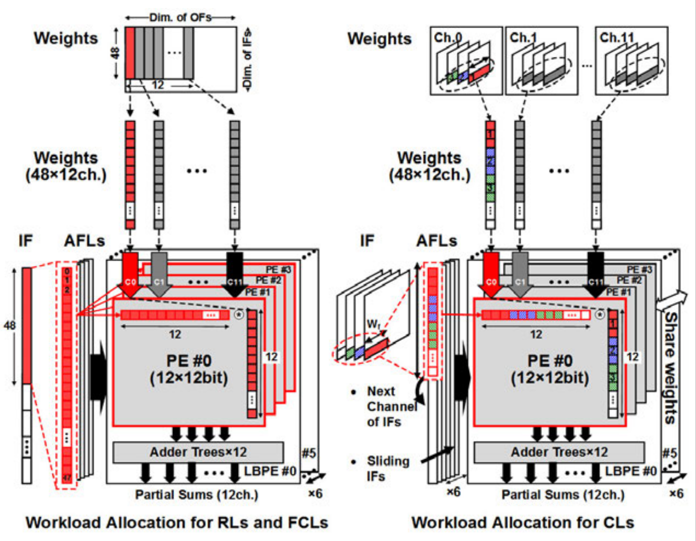
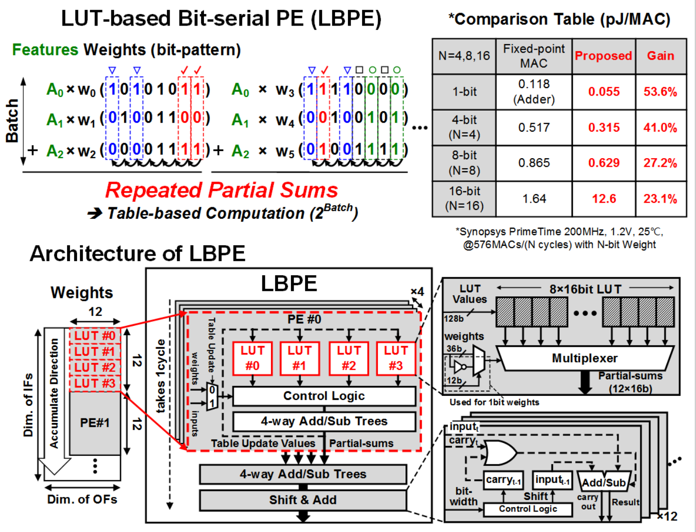

Workload Allocation for RLs and FCLs
Workload Allocation for CLs
PE #0 (12×12bit)
Adder Trees×12
Partial Sums (12ch.)
Share weights

Figure 13.3.3: Workload allocation on the unified DNN core.

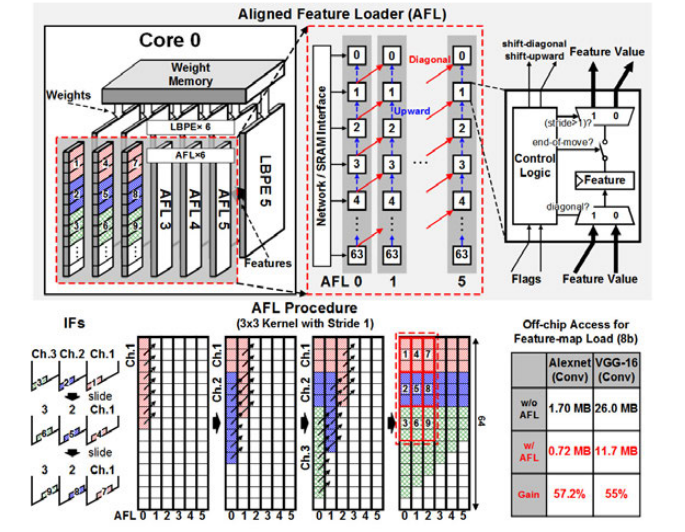


*Comparison Table (pJ/MAC)

N=4,8,16	Fixed-point MAC	Proposed	Gain
1-bit	0.118 (Adder)	0.055	53.6%
4-bit (N=4)	0.517	0.315	41.0%
8-bit (N=8)	0.865	0.629	27.2%
16-bit (N=16)	1.64	12.6	23.1%

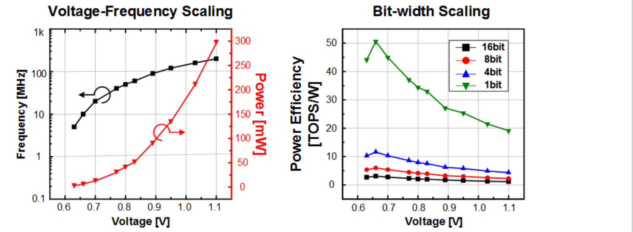
*Synopsys PrimeTime 200MHz, 1.2V, 25℃, @576MACs/(N cycles) with N-bit Weight

Figure 13.3.4: LUT-based bit-serial processing elements.


Aligned Feature Loader (AFL)
AFL Procedure (3x3 Kernel with Stride 1)
Off-chip Access for Feature-map Load (8b)

	Alexnet (Conv)	VGG-16 (Conv)
w/o AFL	1.70 MB	26.0 MB
w/ AFL	0.72 MB	11.7 MB
Gain	57.2%	55%

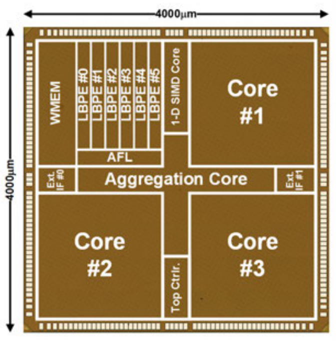
Figure 13.3.5: Aligned feature loader for reduction of off-chip memory accesses.



	ISSCC2017 [1]		ISSCC2017 [2]		S.VLSI2017 [5]		S.VLSI2017 [6]		This Work	
Purpose	CLs, RLs, FCLs		CLs		CLs, RLs, FCLs		RLs, FCLs		CLs, RLs, FCLs	
Process [nm]	65		28		65		65		65	
Die Area [mm²]	16		1.87		19.36		12		16	
Supply Voltage [V]	0.77-1.1		0.605-1.05		0.67-1.2		0.55-1.0		0.63-1.1	
Max. Freq. [MHz]	200		200		200		400		200	
SRAM [KB]	290		144		348		153		256	
PE Bit-precision [bit]	4, 8, 16		4, 8, 16		4, 8, 16		1, 2 (Binarized Activation)		1-16 (Continuous Activation)	
Peak Performance [GOPS]	1200(4b) 300(16b)		408(4b) 76(16b)		409.6 (4b)		1380(1b)		7372(1b), 1382(4b), 691.2(8b), 345.6(16b)	
Energy Efficiency [TOPS/W]	@50MHz 0.77V	8.1 (4b)	@50MHz 0.63V	10.0[1] (4b)	@10MHz 0.67V	5.09 (4b)	@100MHz 0.55V	6.0 (1bx1b)	3.08 (16b)[2]	
									11.6 (4b)[3]	
		2.1 (16b)	@100MHz 0.70V	0.53 (16b)	@200MHz 1.2V	1.06	@400MHz 1.0V	2.3 (1bx1b)	50.6 (1b)[4]	
Power Consumption [mW]	@50MHz 0.77V	34.6	@50MHz 0.63V	7.5	@10MHz 0.67V	4	@100MHz 0.55V	50	@5MHz 0.63V	3.2
	@200MHz 1.1V	279	@200MHz 1.05V	300	@200MHz 1.2V	447	@400MHz 1.0V	600	@200MHz 1.1V	297

1) The CNN sparsity was exploited.
2) DNN Core: 10MHz, Link: 10MHz, 0.66V
3) DNN Core: 5MHz, Link: 10MHz, 0.66V
4) DNN Core: 2MHz, Link: 10MHz, 0.66V

Figure 13.3.6: Measurement results and performance comparison table.



	Specifications	
Technology	65nm 1P8M Logic CMOS	
Die Area	4000×4000µm^2	
SRAM	256KB	
Supply @ (Nominal)	1.1V	
Frequency (Nominal)	200MHz	
Weight Bit-precision	1 - 16	
Power Consumption [mW]	297 @200MHz, 1.1V	
	3.2 @5MHz, 0.63V	
Power Efficiency [TOPS/W]	CLs	50.6 (1b)
		11.6 (4b)
		5.57 (8b)
	*FCLs, RLs	46.7 (1b)
		10.3 (4b)
		5.32 (8b)

* Evaluated on 512x512 Weights

Figure 13.3.7: Chip micrograph and performance summary.

13.4 A 9.02mW CNN-Stereo-Based Real-Time 3D Hand-Gesture Recognition Processor for Smart Mobile Devices

Sungpill Choi, Jinsu Lee, Kyuho Lee, Hoi-Jun Yoo

KAIST, Daejeon, Korea

Recently, 3D hand-gesture recognition (HGR) has become an important feature in smart mobile devices, such as head-mounted displays (HMDs) or smartphones for AR/VR applications. A 3D HGR system in Fig. 13.4.1 enables users to interact with virtual 3D objects using depth sensing and hand tracking. However, a previous 3D HGR system, such as Hololens [1], utilized a power consuming time-of-flight (ToF) depth sensor (>2W) limiting 3D HGR operation to less than 3 hours. Even though stereo matching was used instead of ToF for depth sensing with low power consumption [2], it could not provide interaction with virtual 3D objects because depth information was used only for hand segmentation. The HGR-based UI system in smart mobile devices, such as HMDs, must be low power consumption (<10mW), while maintaining real-time operation (<33.3ms). A convolutional neural network (CNN) can be adopted to enhance the accuracy of the low-power stereo matching. The CNN-based HGR system comprises two 6-layer CNNs (stereo) without any pooling layers to preserve geometrical information and an iterative-closest-point/particle-swarm optimization-based (ICP-PSO) hand tracking to acquire 3D coordinates of a user's fingertips and palm from the hand depth. The CNN learns the skin color and texture to detect the hand accurately, comparable to ToF, in the low-power stereo matching system irrespective of variations in external conditions [3]. However, it requires >1000 more MAC operations than previous feature-based stereo depth sensing, which is difficult in real-time with a mobile CPU, and therefore, a dedicated low-power CNN-based stereo matching SoC is required.

In this paper, we describe an accurate, low power (<10mW), and real-time (<33.3ms) 3D HGR processor for smart mobile devices with 3 key features: 1) a pipelined CNN processing element (PE) with a shift MAC operation for high throughput by maximizing core utilization; 2) triple ping-pong buffers with workload balancing for fast line streaming by reducing external accesses; and 3) nearest-neighbor searching (NNS) processing-in-memory (PIM) for high energy efficiency by reducing the number of bitlines requiring pre-charge in SRAM.

Figure 13.4.2 shows the overall architecture of the HGR processor that consists of a CNN-stereo engine (CSE) and an ICP-PSO engine (IPE). The CSE contains two line-streaming CNN cores with 4 locally distributed memories and one matching core. The CNN core has one pipelined CNN PE and a local DMA with a forwarding/backwarding (FWD/BWD) unit to balance workloads between the CNN cores. The IPE consists of a NNS unit with 16-way parallel NNS PIMs and a hand-tracking unit.

Figure 13.4.3 shows the pipelined CNN PE architecture with shift MAC operation, which performs 1D convolution, for a line-streaming CNN. The entire 2-D convolution is performed by repetition of shift MAC. The shift MAC operation with a 3×3 filter in Fig. 13.4.3 consists of three stages: 1) shifting feature maps and filters, 2) element-wise multiplication, and 3) partia-sum accumulation. First, input feature maps and filters are loaded into shift registers and both are shifted by 1-index in every clock cycle. Then, the active weights of each channel are multiplied with active features element-by-element. Finally, multiplication results are accumulated to obtain 1D convolution results in 3 cycles. The line-streaming CNN operation is accelerated by the 7-stage pipelined CNN PE that processes 48 MACs per cycle with 96% core utilization. Moreover, the pipelined architecture enables line-streaming processing, as well as memory access latency hiding to achieve 1.80TOPS/W, 60MHz at 0.9V.

Memory management of the CNN core is shown in Fig. 13.4.4. The hardware utilizes triple ping-pong memories to store feature maps, where each memory is accessed simultaneously to feed pipeline inputs, write back pipeline outputs, and to access an external interface, respectively. Instead of storing the entire feature maps on the chip, the line-streaming processing with only 3-to-5 lines of feature maps reduces 90.1% of required data that must be fetched from/to off-chip. As a result, the triple ping-pong operation hides the external access time behind CNN computation, and the hand tracking system does not need external accesses to fetch intermediate feature maps. In addition, the FWD/BWD unit balances workloads between two CNN cores automatically. As shown in top-right of Fig. 13.4.4, data in each core becomes unbalanced due to the reduced size of feature maps after convolution, especially in a distributed memory architecture. The data transaction time between CNN cores must be well defined to balance their workloads [4]. The FWD/BWD units keep CNN core workloads identical throughout CNN processing and, as shown in Fig. 13.4.4, exchange feature-map boundary data with one another when local feature maps are fetched. Moreover, the internal data transaction time for workload balancing can be hidden behind the CNN pipeline. As a result, the triple ping-pong buffers with the FWD/BWD unit reduce overall CNN processing time by 23.9%

Figure 13.4.5 shows the PIM architecture specialized for NNS to track a user's hands in the IPE. Hand tracking requires >360K-node k-d tree NNS between the 46-sphere model in the memory and the depth input from the CSE. In the proposed PIM, NNS operation is composed of 2 half cycles such that it performs NNS on a parent in the first half cycle, 1 read operation for fetching next parent address, and on a selected child in the remaining half cycle. The proposed PIM is composed of 4 cell arrays (CA) for a parent node, two child nodes, and a next searching address. Each CA has 36×16 8T-SRAM cells and the bitlines are separated as read bitlines (RBLs) and write bitline (WBLs). The 3 CAs for a parent and 2 child nodes contain ripple-carry comparators that output "C_{OUT} = Sign(WBL-RBL) **OR** C_{IN}" to the comparison bitline (CBL). A parent CBL is connected to an address decoder to activate the selected child CA by changing LSB of the address decoder in later half cycle of NNS. The proposed PIM achieves 6× speed-up compared with the conventional SRAM design, which requires 6 operations to complete NNS. Moreover, it can skip read operations by 1/3 so that redundant pre-charging power consumption on global bitlines can be reduced by 63.9% (2.8× energy-efficiency enhancement).

Measurement results of hand-depth sensing and hand tracking in the 3D HGR processor are shown in Fig. 13.4.6. Thanks to the CNN stereo, the processor can acquire accurate hand depth showing distinguishable depth of fingertips and low disparity error. In 20cm-to-40cm active range, the average hand tracking accuracy is 4.3mm with 5cm separated VGA stereo cameras, and it achieves mm-scale accuracy of 3D HGR. In addition, the proposed processor consumes 9.02mW @ 50MHz, 0.85V for real-time 1-hand 3D HGR, which is 14× less than the state-of-the-art UI processor [2]. As a result, the 3D HGR processor can satisfy the required power budget (<10mW) with 30ms latency. Moreover, the CNN-stereo engine achieves 1.80TOPS/W which is 1.45× more energy efficient than a state-of-the-art distributed memory architecture [4]. The line-streaming CNN architecture uses 781.5KB on-chip memory, while [4-6] needed ~MB memory, which is impossible to realize as on-chip memory.

The 3D HGR processor for smart mobile devices is fabricated in 65nm CMOS technology, and it occupies 4×4mm^2 integrating 781.5KB of SRAM. Its maximum and average hand tracking error are 10.6mm and 4.3mm, respectively, where that of a ToF system is ~5mm pm average. The highly accurate 3D HGR processor consumes only 9.02mW with 30ms system latency.

References:

[1] Hololens Hardware detail. Available: https://developer.microsoft.com/en-us/windows/mixed-reality/hololens_hardware_details
[2] S. Park, et al., "A 126.1mW Real-Time Natural UI/UX Processor with Embedded Deep-Learning Core for Low-Power Smart Glasses," *ISSCC*, pp. 254-255, 2016.
[3] W. Luo, et al., "Efficient Deep Learning for Stereo Matching," *CVPR*, pp. 5695-5703, 2016.
[4] K. Bong, et al., "A 0.62mW Ultra-Low-Power Convolutional-Neural-Network Face-Recognition Processor and a CIS Iintegrated with Always-On Haar-Like Face Detector," *ISSCC*, pp. 248-249, 2017.
[5] Y. H. Chen, et al., "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," *ISSCC*, pp. 262-263, 2016.
[6] D. Shin, et al., "DNPU: An 8.1TOPS/W Reconfigurable CNN-RNN Processor for General-Purpose Deep Neural Networks," *ISSCC*, pp. 240-241, 2017.

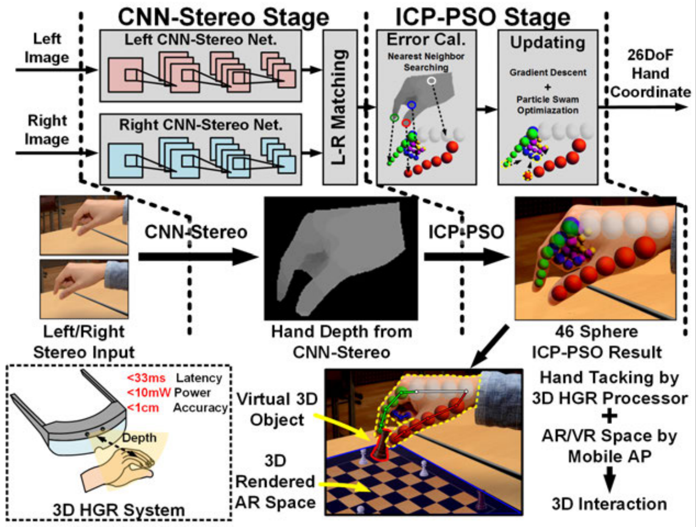


Figure 13.4.1: 3D hand-gesture recognition in mobile smart devices.

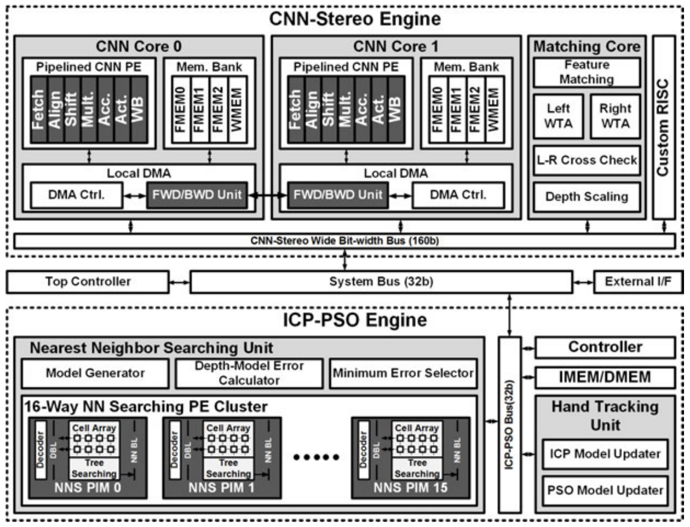


Figure 13.4.2: Overall architecture.

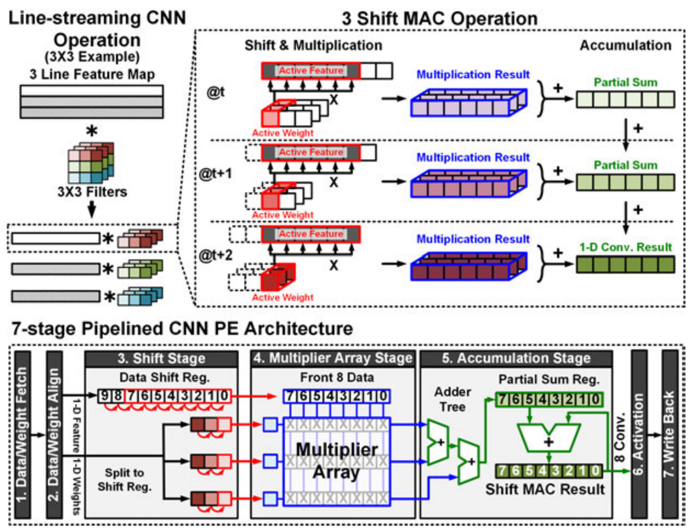


Figure 13.4.3: Pipelined CNN PE architecture with shift MAC.

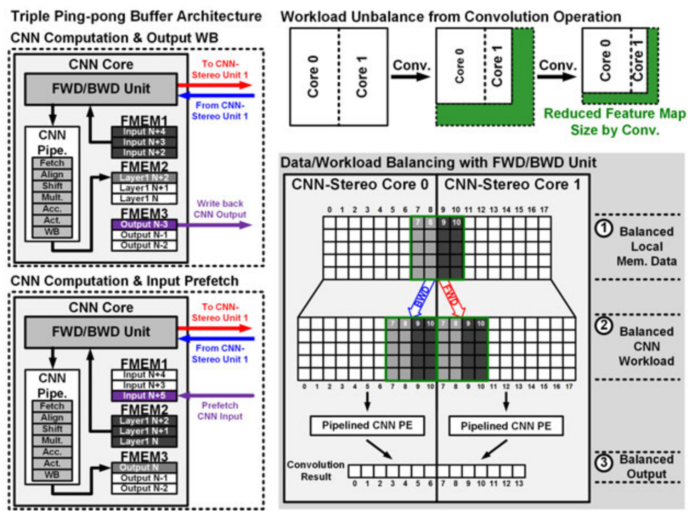


Figure 13.4.4: Triple ping-pong buffer architecture with workload balancing.

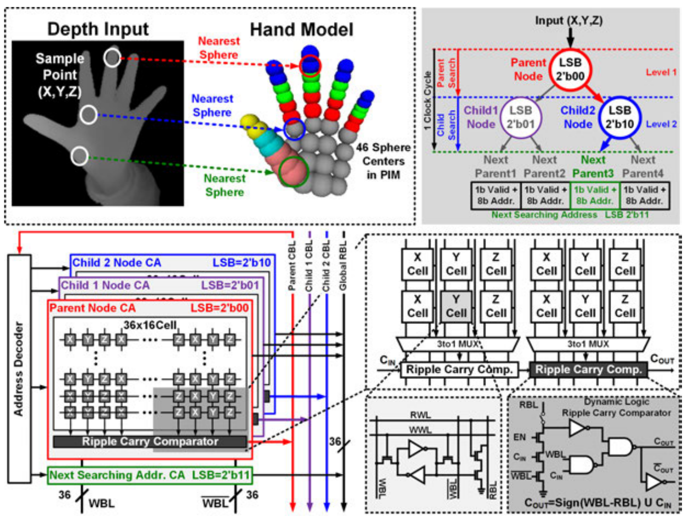
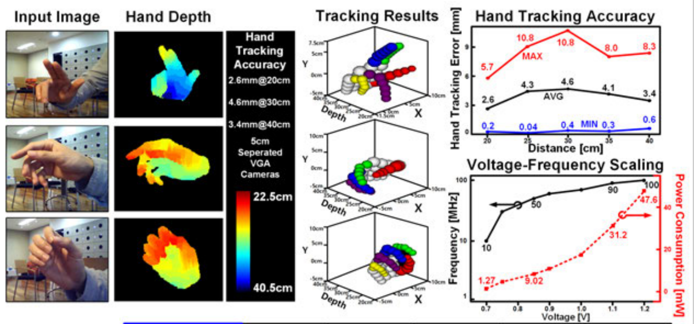
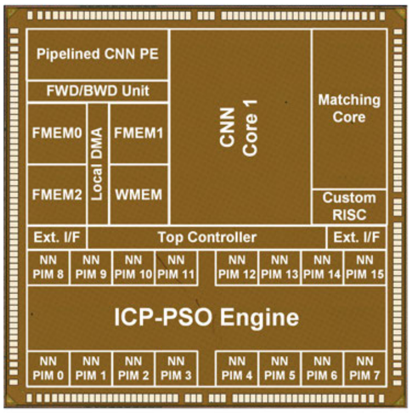


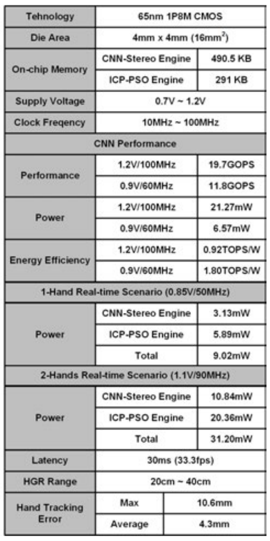
Figure 13.4.5: Nearest-neighbor searching processing-in-memory (PIM).



	This Work	[1] Hololens	[2] ISSCC'16	[4] ISSCC'17	[5] ISSCC'16	[6] ISSCC'17
Tehnology	65nm	N/A	65nm	65nm	65nm	65nm
Power	9.02mW @ 1 hand	3000mW	126.1mW	2.35mW	278mW	29mW
Real-time 3D HGR	O	O(Software)	X	X	X	X
Average Error of 3D HGR	4.3mm @20~40cm	5mm @15~45cm	N/A	N/A	N/A	N/A
Depth Sensing Method	Stereo Matching	Time-of-Flight(ToF)	Stereo Matching	N/A	N/A	N/A
		CNN Performance (16b MAC)				
Power	6.57mW	N/A	N/A	2.35mW	278mW	29mW
Peak Performance	11.8GOPS	N/A	N/A	2.918GOPS	84.0GOPS	235GOPS
Energy Efficiency	1.80TOPS/W	N/A	N/A	1.24TOPS/W	0.99TOPS/W	2.1TOPS/W

Figure 13.4.6: Measurement results and comparison table.





Tehnology	65nm 1P8M CMOS	
Die Area	4mm x 4mm (16mm^2)	
On-chip Memory	CNN-Stereo Engine	490.5 KB
	ICP-PSO Engine	291 KB
Supply Voltage	0.7V ~ 1.2V	
Clock Freqency	10MHz ~ 100MHz	
CNN Performance		
Performance	1.2V/100MHz	19.7GOPS
	0.9V/60MHz	11.8GOPS
Power	1.2V/100MHz	21.27mW
	0.9V/60MHz	6.57mW
Energy Efficiency	1.2V/100MHz	0.92TOPS/W
	0.9V/60MHz	1.80TOPS/W
1-Hand Real-time Scenario (0.85V/50MHz)		
Power	CNN-Stereo Engine	3.13mW
	ICP-PSO Engine	5.89mW
	Total	9.02mW
2-Hands Real-time Scenario (1.1V/90MHz)		
Power	CNN-Stereo Engine	10.84mW
	ICP-PSO Engine	20.36mW
	Total	31.20mW
Latency	30ms (33.3fps)	
HGR Range	20cm ~ 40cm	
Hand Tracking Error	Max	10.6mm
	Average	4.3mm

Figure 13.4.7: Chip photography and performance summary.

13.5 An Always-On 3.8μJ/86% CIFAR-10 Mixed-Signal Binary CNN Processor with All Memory on Chip in 28nm CMOS

Daniel Bankman[1], Lita Yang[1], Bert Moons[2], Marian Verhelst[2], Boris Murmann[1]

[1]Stanford University, Stanford, CA; [2]KU Leuven, Leuven, Belgium

The trend of pushing deep learning from cloud to edge due to concerns of latency, bandwidth, and privacy has created demand for low-energy deep convolutional neural networks (CNNs). The single-layer classifier in [1] achieves sub-nJ operation, but is limited to moderate accuracy on low-complexity tasks (90% on MNIST). Larger CNN chips provide dataflow computing for high-complexity tasks (AlexNet) at mJ energy [2], but edge deployment remains a challenge due to off-chip DRAM access energy. This paper describes a mixed-signal binary CNN processor that performs image classification of moderate complexity (86% on CIFAR-10) and employs near-memory computing to achieve a classification energy of 3.8μJ, a 40× improvement over TrueNorth [3]. We accomplish this using (1) the BinaryNet algorithm for CNNs with weights and activations constrained to +1/-1 [4], which drastically simplifies multiplications (XNOR) and allows integrating all memory on-chip; (2) an energy-efficient switched-capacitor (SC) neuron that addresses BinaryNet's challenge of wide vector summation; (3) architectural parallelism, parameter reuse, and locality.

Figure 13.5.1 illustrates the function and network topology of our design. By enforcing structural regularity, we allow the physical architecture to maximally exploit the locality of the CNN algorithm. Each CNN layer carries out a multi-channel, multi-filter convolution. The number of filters in each convolutional layer is restricted to 256, the filter size is 2×2, and the number of channels is 256. The circuit benefits of this regularity are short wires and arrayed, low fan-out de-multiplexers, which minimize path loading between memory and logic.

Figure 13.5.2 shows the top-level architecture, which supports up to 9 layers with a customized instruction set for input-output actions, CNN and fully-connected (FC) layers. The processor reads an RGB image, converts the channels to 85-level thermometer codes, and stacks them into a 256-channel image as the CNN input. At the output, an FC layer digitally computes the 4b class label. For a CNN layer, east and west SRAM banks alternate roles between input and output in a ping-pong fashion. These SRAM banks are 256b wide, each word representing a 256-channel pixel. Computation of a filter is completed inside a neuron, eliminating partial sums. The weights are transferred from SRAM to local neuron memory (latches) and reused, while the filter traverses the image. A data-parallel array of 64 neurons processes a patch of the input image, amortizing the input image SRAM read energy per filter computation by 64×. The input DEMUX block interfaces between SRAM (which loads a pixel) and the neuron array (which receives a patch). For the FC layer, weights are loaded from a separate SRAM bank 64 channels at a time, and the multiply-accumulate operation is performed sequentially in the digital domain.

Figure 13.5.3 shows how locality translates to reduced loading. The input DEMUX is an array of 1-to-4 de-multiplexers with output registers. Each pixel of the input image can be reused in the processing of two overlapping patches, amortizing the input image SRAM read energy per filter computation by 2×. A 2-by-2 crossbar interchanges pixel pairs at the neuron array input. Filter weights are transferred over a 4b per neuron bus, split into north and south halves to reduce the loading of weight transfers by 2×. To minimize neuron array to memory wiring, each neuron writes to the same 4 output channels (1 per filter group) in each CNN layer, allowing implementation of the output DEMUX block as an array of 1-to-4 de-multiplexers. Max pooling occurs incrementally during convolution by first reading a bit in the output image SRAM, and then writing back its logical OR with a neuron output.

Figure 13.5.4 shows the neuron schematic, which computes the weighted sum of a filter with a patch of the input image. With memory energy amortized by parallelism and reuse, and multiplication reduced to XNOR, high-fan-in addition becomes the main bottleneck. However, in the employed SC neuron, the energy cost of addition is reduced by the small voltage swing at the charge conservation node. In contrast, a digital adder tree would involve rail-to-rail swings along its stages and exhibits a larger amount of switched capacitance. The neuron's dominant noise source is the comparator, but its energy cost is amortized over 1024 weights and the CNN can tolerate some noise. As a result, the SC neuron is amenable to low-voltage operation, and uses a combined 0.6V digital supply/analog reference and a 0.8V comparator supply. Because the SC neuron performs a weighted sum with data-dependent switching (apart from the comparator), its energy scales with activity, like static CMOS. The SC neuron uses a capacitive DAC (CDAC) with four sections: a 1024b thermometer section for applying a filter, a binary-weighted section for the neuron's bias, a threshold section (comparator), and a common-mode (CM) setting section to compensate for parasitics at the charge conservation node. Comparator offset is digitized using calibration at startup, stored in a local register, and subtracted from the bias loaded from SRAM during weight transfer. In environments where large temperature changes may induce significant offset drift, calibration can be performed periodically (e.g. once per second) at negligible cost in average energy per classification and throughput. Behavioral Monte Carlo simulations were run to determine the amount of comparator noise, offset, and unit-capacitor mismatch that the CNN can tolerate without degradation in classification accuracy, resulting in a comparator designed for 4.6mV offset and a 1fF unit capacitor. Because the voltage representing the weighted sum is developed at the charge conservation node, top and bottom plate parasitics do not affect linearity. During convolution, the CDAC is periodically cleared (sampling 0V) as required by leakage at the top plates. To prevent drawing excessive charge from the supply, the unit-capacitor bottom plate nodes are discharged by asserting CLR before the top plate is discharged via CLR_e. To prevent asymmetric charge injection, the top plate switches are opened before the bottom plate voltages resume their values set by filter weights, image inputs, and biases.

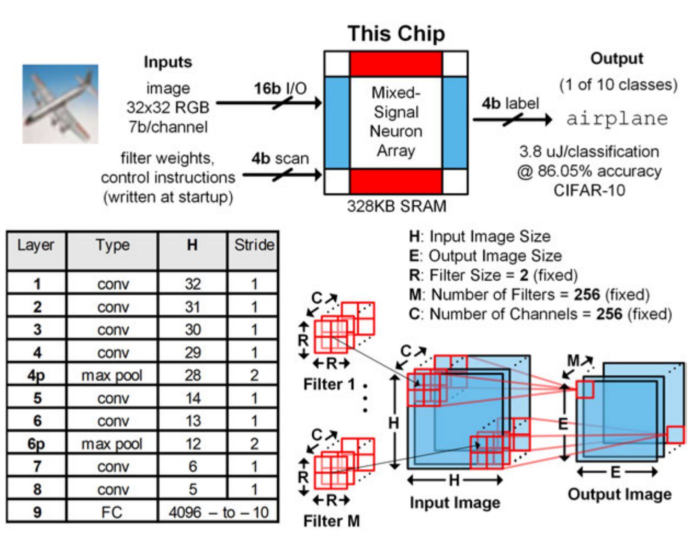
Figure 13.5.5 shows the measured results at room temperature. Ten different chips were measured to evaluate the accuracy spread due to thermal noise and mismatch in the SC neuron. At nominal supply voltages ($V_{DD}=V_{MEM}=1.0V$, $V_{NEU}=0.6V$, $V_{COMP}=0.8V$), the chips operate up to 380frames/s (FPS) and achieve 5.4μJ/classification. Lowering V_{DD} and V_{MEM} to 0.8V leads to 3.8μJ/classification (1.43× reduction) at 237 FPS. The mean classification accuracy is 86.05% (see histogram), the same as observed in a perfect digital model. The histogram spread is solely caused by the noise and mismatch in the SC neuron (which can notably lead to a higher classification accuracy than in the perfect digital model). The 95% confidence interval in mean classification accuracy is 86.01% to 86.10%, measured over 10 chips, 30 runs each through the 10,000 image CIFAR-10 test set. Not included in these energy figures is the 1.8V chip I/O energy, which amounts to 0.43μJ (a small fraction of the core energy).

To explore further energy savings, we reduced V_{DD} to 0.6V and set V_{MEM} to 0.53V, 0.52V, 0.51V and 0.50V to show the impact of bit errors. The mean accuracy degrades to 85.7%, 85.2%, 84.2% and 82.5%, respectively. The large error bars for the lower voltages (see Fig. 13.5.5) are due to SRAM V_{MIN} variations across the 10 chips. At $V_{DD}=0.6V$ and $V_{MEM}=0.53V$ (borderline practical), the chip consumes 2.61μJ/classification, a 2.1× reduction versus nominal supplies. From the breakdown in Fig. 13.5.5, we see that neuron energy increases due to leakage at the lower FPS imposed by voltage scaling. However, this increase is small compared to the logic and memory savings.

Figure 13.5.6 compares this work with prior art and Fig. 13.5.7 shows a photo of the 2.44mm×2.44mm die. On the same benchmark dataset (CIFAR-10), we achieve 40-60× improvement in energy per classification over [3], which does not exploit locality and thus suffers from high interconnect activity. The binarized DNN accelerator in [5] has all memory on-chip, but cannot exploit weight reuse, attaching the energy cost of an SRAM bit load with each XNOR operation. The spiking LCA network in [6] exhibits low energy, but has a relatively low accuracy for a lower-complexity task (MNIST).

Acknowledgements:
Silicon fabrication was provided by the TSMC university shuttle program. We thank Scott Liao for design support. This work was funded in part by Systems on Nanoscale Information fabriCs (SONIC), one of the six STARnet Centers, sponsored by MARCO and DARPA.

References:
[1] J. Zhang, et al., "A Machine-learning Classifier Implemented in a Standard 6T SRAM Array," *IEEE Symp. VLSI Circuits*, 2016.
[2] Y. H. Chen, et al., "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," *ISSCC*, pp. 262-263, 2016.
[3] S. Esser, et al., "Convolutional networks for fast, energy-efficient neuromorphic computing," *Proc. Natl. Acad. Sci. USA*, vol. 113, no. 41, pp. 11441-11446, 2016.
[4] M. Courbariaux, et al., "Binarized Neural Networks: Training Neural Networks with Weights and Activations Constrained to +1 or -1," *arXiv preprint*: 1602.02830v3, 2016.
[5] K. Ando, et al., "BRein Memory: A 13-Layer 4.2 K Neuron/0.8 M Synapse Binary/Ternary Reconfigurable in-Memory Deep Neural Network Accelerator in 65 nm CMOS," *IEEE Symp. VLSI Circuits*, 2017.
[6] F. Buhler, et al., "A 3.43TOPS/W 48.9pJ/Pixel 50.1nJ/Classification 512 Analog Neuron Sparse Coding Neural Network with On-Chip Learning and Classification in 40nm CMOS," *IEEE Symp. VLSI Circuits*, 2017.



Layer	Type	H	Stride
1	conv	32	1
2	conv	31	1
3	conv	30	1
4	conv	29	1
4p	max pool	28	2
5	conv	14	1
6	conv	13	1
6p	max pool	12	2
7	conv	6	1
8	conv	5	1
9	FC	4096 – to – 10	

Figure 13.5.1: System design and binary CNN topology.

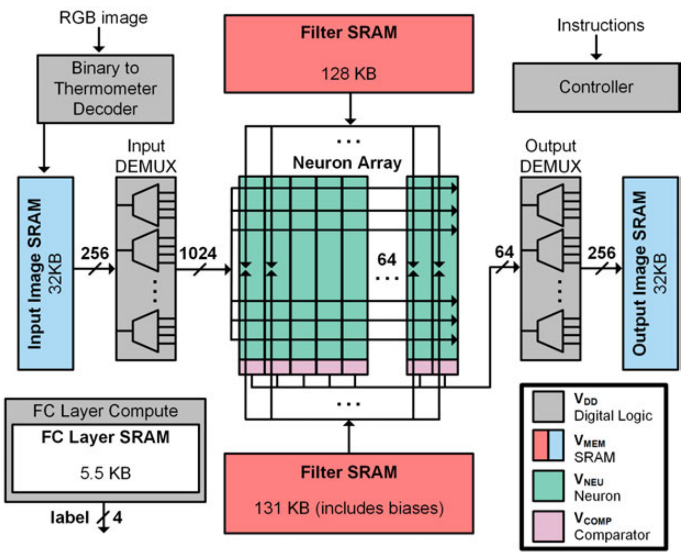
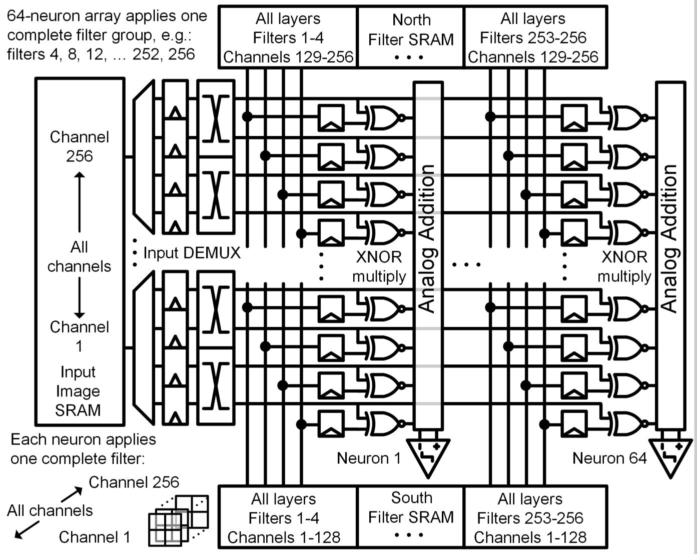


Figure 13.5.2: Top-level architecture with 64 neurons.



64-neuron array applies one complete filter group, e.g.: filters 4, 8, 12, … 252, 256
All layers Filters 1-4 Channels 129-256
North Filter SRAM
All layers Filters 253-256 Channels 129-256
Channel 256
All channels
Channel 1
Input Image SRAM
Input DEMUX
XNOR multiply
Analog Addition
Neuron 1
Neuron 64
Each neuron applies one complete filter:
Channel 256
All channels
Channel 1
All layers Filters 1-4 Channels 1-128
South Filter SRAM
All layers Filters 253-256 Channels 1-128

Figure 13.5.3: Locality in logic design and physical architecture.

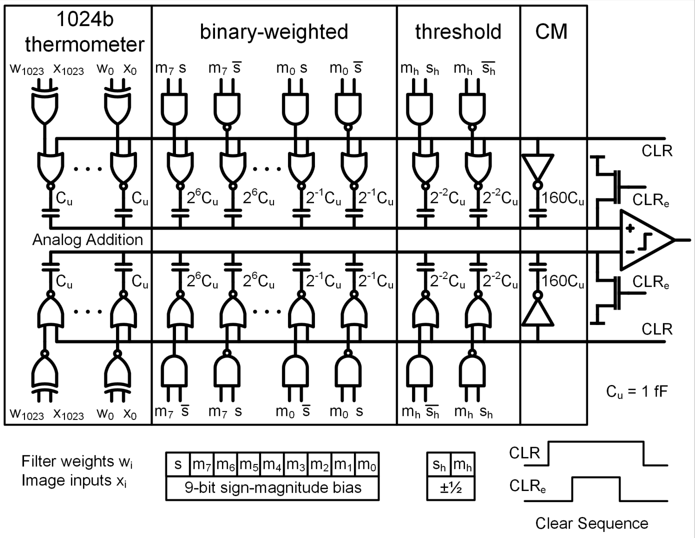


Figure 13.5.4: Switched-capacitor neuron using charge redistribution for wide vector summation.

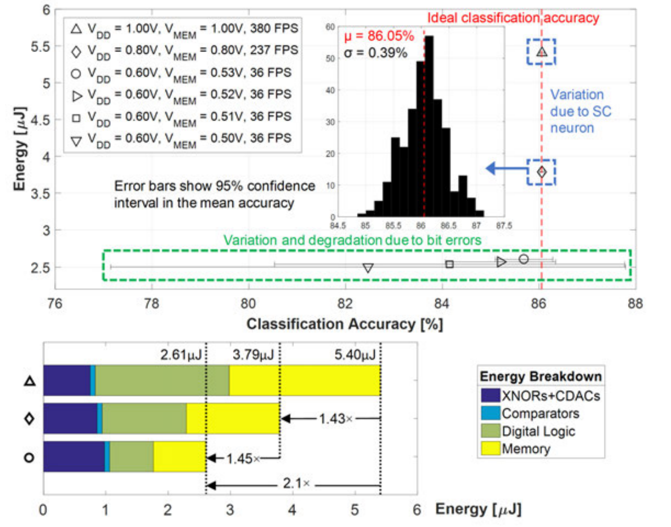


Figure 13.5.5: Measured energy and accuracy on CIFAR-10 image classification.

	This work		[3] IBM TrueNorth	[5] VLSI '17	[6] VLSI '17
Technology	28nm		28nm	65nm	40nm
Algorithm	CNN		CNN	DNN	LCA
Dataset	CIFAR-10		CIFAR-10	MNIST	MNIST
(Weight, Activation) Precision [bits]	(1, 1)		(1.6, 1)	(1.6, 1)	(4, 1)
Supply [V]	V_{NEU} = 0.6 V_{COMP} = 0.8 V_{DD} = 0.8 V_{MEM} = 0.8	V_{NEU} = 0.6 V_{COMP} = 0.8 V_{DD} = 0.6 V_{MEM} = 0.53	1.0	0.55 – 1.0	0.9
Classification Accuracy [%]	86.05	85.69	83.41	90.1	88
Energy per Classification [uJ]	3.79	2.61	164	0.28 – 0.73	0.050
Power [mW]	0.899	0.094	204.4	50 – 600	87
Frame Rate [FPS]	237	36	1249	820K – 3280K	1.7M
Arithmetic Energy Efficiency	532 1b-TOPS/W	772 1b-TOPS/W	–	6.0 – 2.3 TOPS/W	3.43 TOPS/W

Figure 13.5.6: Comparison to state of the art.

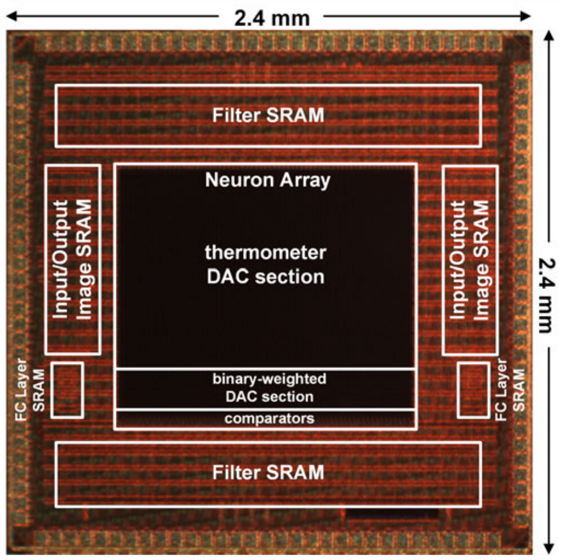


Figure 13.5.7: Die photo.

13.6 A 1.8Gb/s 70.6pJ/b 128×16 Link-Adaptive Near-Optimal Massive MIMO Detector in 28nm UTBB-FDSOI

Wei Tang[1], Hemanth Prabhu[2], Liang Liu[2], Viktor Öwall[2], Zhengya Zhang[1]

[1]University of Michigan, Ann Arbor, MI
[2]Lund University, Lund, Sweden

This work presents a 2.0mm^2 128×16 massive MIMO detector IC that provides 21dB array gain and 16× multiplexing gain at the system level. The detector implements iterative expectation-propagation detection (EPD) for up to 256-QAM modulation. Tested with measured channel data [1], the detector achieves 4.3dB processing gain over state-of-the-art massive MIMO detectors [2, 3], enabling 2.7× reduction in transmit power for battery-powered mobile terminals. The IC uses link-adaptive processing to meet a variety of practical channel conditions with scalable energy consumption. The design is realized in a condensed systolic array architecture and an approximate moment-matching circuitry to reach 1.8Gb/s at 70.6pJ/b. The performance and energy efficiency can be tuned over a wide range by UTBB-FDSOI body bias.

Real-time detection for massive MIMO is compute-intensive and power-hungry due to massive matrix dimensions and fast varying channels. Previous works [2, 3] demonstrated low-complexity massive MIMO detectors based on independent and identically distributed (i.i.d.) channel assumption in massive MIMO. The i.i.d. channel assumption is impractical and these simplified detectors suffer from significant performance losses when tested in measured massive MIMO channels, especially in cases of high user load. In designing a practical massive MIMO detector, we select EPD that leverages iterative interference cancellation [4] to offer near-optimal performance even in unfavorable channel conditions, while its complexity is limited to $O(K^3)$ per iteration, where K is the number of users. We exploit EPD's iterative processing to adapt the processing effort to the channel so to achieve the required BER at the lowest energy. The EPD design incorporates explicit matrix inversion, so it could be reused for both uplink and downlink processing. Evaluated using measured massive MIMO channels, the EPD outperforms a linear MMSE detector by 0.7dB, 4.3dB, and 3.5dB in i.i.d., non-line-of-sight (NLOS), and line-of-sight (LOS) conditions, respectively, as shown in Fig. 13.6.1.

The EPD architecture is shown in Fig. 13.6.2. The Gram and y^{MF} memory buffers incoming channel and match-filtered uplink streams, and the memory supports flexible access patterns required for reconfiguration. The MMSE parallel interference cancellation (MMSE-PIC) filter cancels the inter-user interference from the uplink user data. The moment-matching unit refines the symbol estimates by incorporating constellation information. The detection-control unit dynamically adjusts the per-iteration processing effort and detects early convergence. Updated symbol estimates from the moment-matching unit are buffered in the symbol estimate memory and fed back to the MMSE-PIC filter for iterative refinement. The architecture is configurable to support vector processing of different lengths to facilitate dynamic dimension reduction, i.e., when a batch of estimates are determined to be reliable, they are frozen and the corresponding users are removed from future iterations. In Fig. 13.6.2, dynamic dimension reduction enables 40-90% of complexity reduction due to the reduced number of users and iteration count. With appropriate threshold choices, the possibility of premature freezing is minimized and the SNR loss is negligible. In designing the silicon prototype, we combine this adaptive architecture with coarse-grained clock gating to save 49.3% power.

One of the most compute-intensive and accuracy-critical parts of the EPD is the matrix inversion block in the MMSE-PIC filter. A systolic array is often used to implement the LDL decomposition to realize highly accurate matrix inversion. The systolic array architecture features a regular architecture, efficient routing and simple control. However, the hardware utilization of a systolic array architecture is only 33.3% [5] due to the need for zero-padding inputs. In this work, we implement a condensed LDL systolic array, which merges under-utilized PE circuitry to improve the hardware utilization to 90% for a 16×16 array, while reducing the interconnect overhead by more than 70%. As shown in Fig. 13.6.3, a PE in a regular systolic array performs division (PE0), multiplication (PE1) or MAC (PE2 and 3) operations and passes its output to the neighboring PEs. In our condensed systolic array, every three PEs in a row are merged. The merging shortens data movements in the systolic array. Rather than passing data along many stages of unused operations in a systolic array, our condensed array limits data movements using holding buffers to maximize data reuse. The data reuse is especially advantageous in our design, as it requires a relatively long 28b data bit width to support a wide range of channel conditions. The condensed array architecture reduces silicon area by 62% compared to the regular systolic array. Moreover, the condensed array shortens data movement delay and dedicates a larger fraction of a clock period to data processing.

The moment-matching unit computes the likelihood of each constellation point to refine the mean and variance of current symbol estimates. The computational complexity is proportional to the product of the modulation size and the number of simultaneously served users. For a 256-QAM, 128×16 massive MIMO system, the complexity is prohibitive. We implement an approximate moment-matching (AMM) circuitry to cut 90% computation by sacrificing a limited, 0.5dB SNR loss. AMM makes the complexity independent of modulation size by exploiting the symmetry of the QAM constellation in computing the mean and variance estimates, thus the approach is favorable in designing a flexible detector that supports a wide range of modulation schemes. Complexity is further reduced with a piecewise linear approximation to compute mean and variance updates: the mean update is reduced to a hard decision of the input soft symbol; and the variance update is fitted into a first-order polynomial function of the input mean and variance. As shown in Fig. 13.6.4, compared to a brute-force moment-matching implementation using 2 dividers, 65 MACs, and 16 exponential evaluations, the AMM circuitry uses only 2 MACs. AMM also eliminates costly exponentiation and division, and reduces intermediate bit width requirements. The technique cuts the silicon area of the moment-matching unit by more than 90%.

An EPD test chip is fabricated in ST 28nm UTBB-FDSOI technology, occupying 2.0mm^2 core area, as shown in Fig. 13.6.7. The measurement results at different core voltages and body biasing in room temperature are shown in Fig. 13.6.5. At a nominal voltage of 1.0V, the EPD chip runs at 512MHz, delivering a system throughput of 1.6Gb/s. By applying forward body biasing of 0.4V, a maximum working frequency of 569MHz is achieved, corresponding to an 11% boost in detection throughput to 1.8Gb/s. The corresponding core power consumption is 127mW, translating to an energy efficiency of 70.6pJ/b. For a low-power application, reverse body biasing of 0.2V and voltage scaling of 0.7V can be applied to reduce the power consumption to 23.4mW at a throughput of 754Mb/s. Compared to the prior MIMO detector designs shown in Fig. 13.6.6, our EPD chip provides flexibility in terms of modulation and channel adaptation, supports both uplink and downlink processing, and achieves a high processing gain, while maintaining competitive energy and area efficiency. Note that the MPD chip in [2] takes advantage of the assumption of the diagonal dominance in i.i.d. channels using a low-complexity, 13b implementation without explicit matrix inversion. However, the MPD encounters an early error floor and fails to provide sufficient processing gain in practical but unfavorable channels such as LOS. In comparison, our EPD chip obtains 4.3dB processing gain in highly correlated channels, equivalent to a 2.7× boost in link margin that can be utilized to significantly lower the TX power and relax the frontend requirements.

Acknowledgements:
The work was supported in part by NSF CCF-1054270, Intel, Silicon Labs, and System Design on Silicon (SoS) Center. Chip fabrication donation was provided by STMicroelectronics. We thank Ove Edfors, Rakesh Gangarajaiah, Babak Mohammadi, Shiming Song and Teyuh Chou for advice.

References:
[1] S. Malkowsky, et al., "The World's First Real-Time Testbed for Massive MIMO: Design, Implementation, and Validation," *IEEE Access*, vol. 5, pp. 9073-9088, 2017.
[2] W. Tang, et al., "A 0.58mm^2 2.76Gb/s 79.8pJ/b 256-QAM Massive MIMO Message-Passing Detector," *IEEE Symp. VLSI Circuits*, pp. 1-2, 2016.
[3] H. Prabhu, et al., "A 60pJ/b 300Mb/s 128×8 Massive MIMO Precoder-Detector in 28nm FD-SOI," *ISSCC*, pp. 60-61, 2017.
[4] J. Céspedes, et al., "Expectation Propagation Detection for High-Order High-Dimensional MIMO Systems," *IEEE Trans. Commun.*, vol. 62, no. 8, pp. 2840-2849, 2014.
[5] S. J. Bellis, et al., "Alternative Systolic Array for Non-Square-Root Cholesky Decomposition," *IEE Proc. Comput. Digit. Technol.*, vol. 144, no. 2, pp. 57-64, 1997.
[6] C.-H. Chen, et al., "A 2.4mm^2 130mW MMSE-Nonbinary LDPC Iterative Detector-Decoder for 4x4 256-QAM MIMO in 65nm CMOS," *ISSCC*, pp. 338-339, 2015.

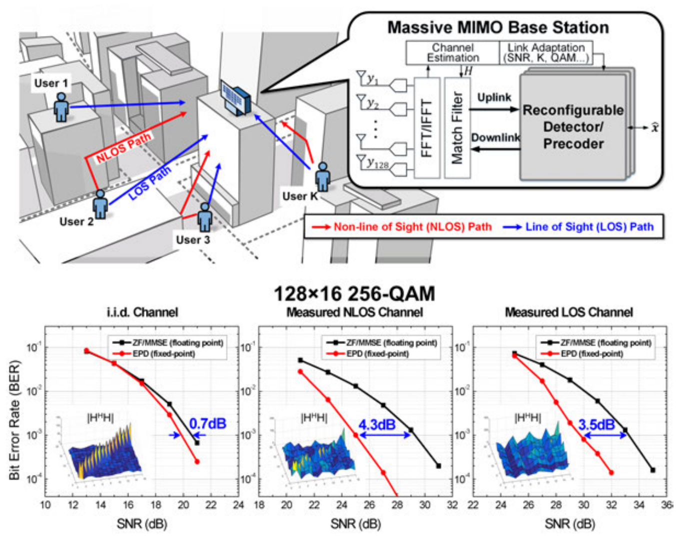


Figure 13.6.1: A multi-user massive MIMO system and BER of different channels. Insets are the Gram matrices $|H^H H|$.

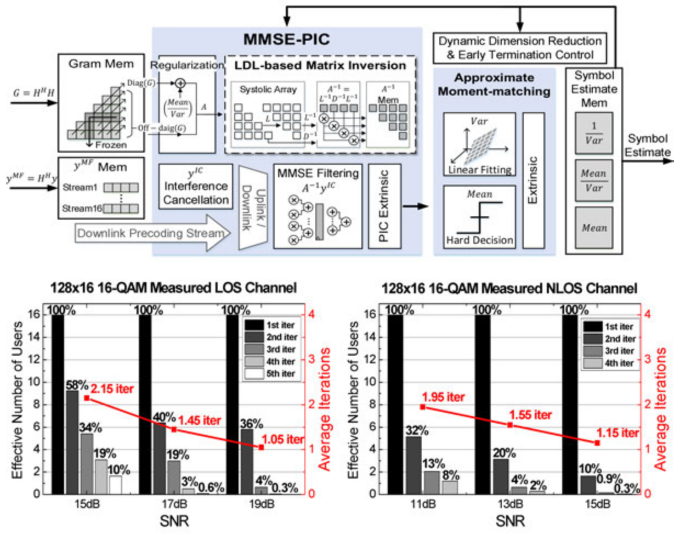


Figure 13.6.2: Link-adaptive EPD architecture and efficiency gains from dimension reduction and early termination.

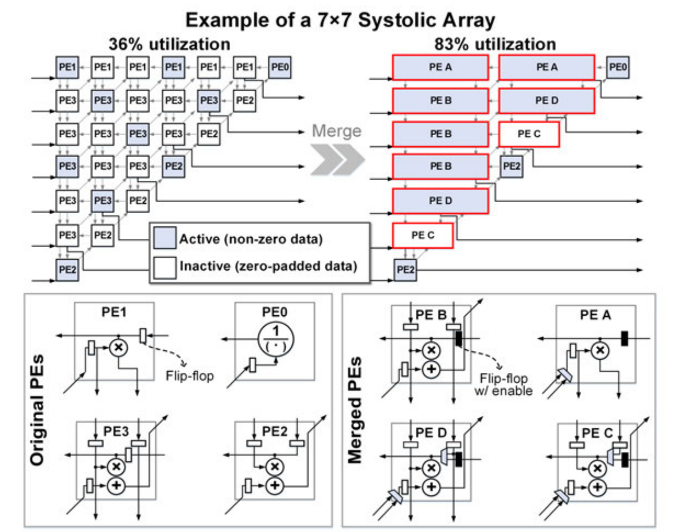
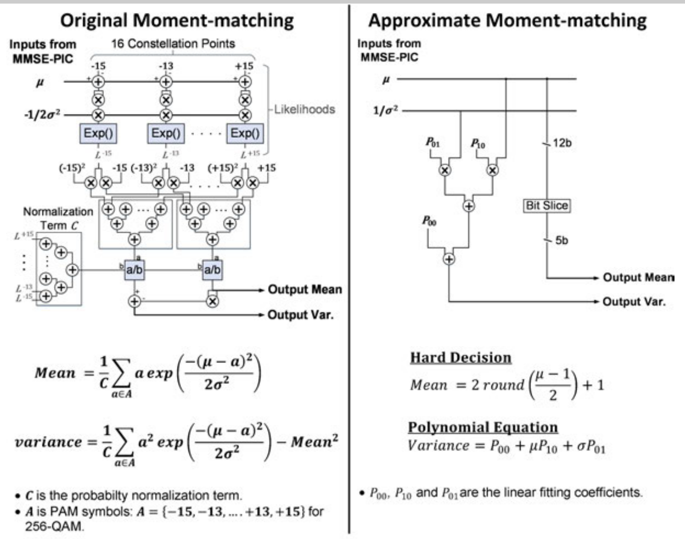


Figure 13.6.3: Condensed LDL systolic array with enhanced utilization and merged PE designs.



Original Moment-matching
Approximate Moment-matching
Inputs from MMSE-PIC
16 Constellation Points
Likelihoods
Normalization Term C
Output Mean
Output Var.
Bit Slice
12b
5b
Hard Decision
Polynomial Equation
C is the probabilty normalization term.
A is PAM symbols: A = {−15, −13, …, +13, +15} for 256-QAM.
P00, P10 and P01 are the linear fitting coefficients.

Figure 13.6.4: Circuitry implementations and complexities of the original and approximate moment-matching.

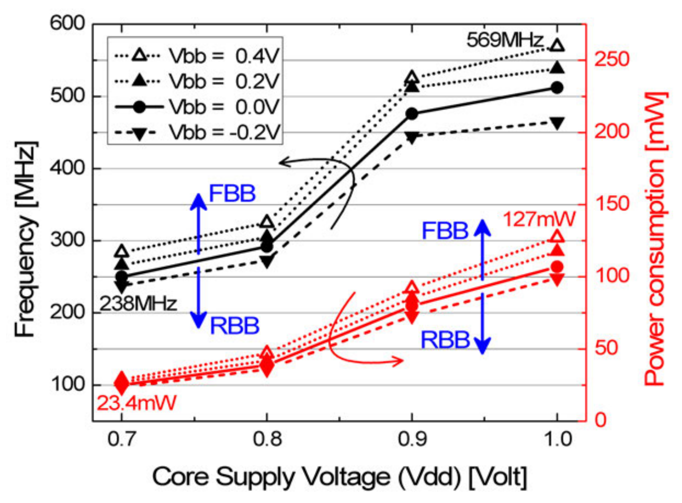


Figure 13.6.5: Measured frequency and power consumption for different core voltages and body biases.

		Chen [6]	Tang [2]	Prabhu [3]	This Work
System	Algorithm	MMSE	MPD [(a)]	MMSE	**EPD [(b)]**
	MIMO [$M \times K$]	4×4	128×32	128×8	**128×16**
	Modulation	256	256	256	**QPSK to 256**
	Channel Adaptiveness	no	no	no	**yes**
	Support Precoding	no	no	yes	**yes**
	Array Gain [dB]	6	21	21	**21**
	Multiplexing Gain	4	32	8	**16**
	Link Margin Improvement [(c)]				
	i.i.d. Channel [dB]	0	1.0	0	**0.6**
	NLOS Channel [dB]	0	Error floor [(d)]	0	**4.3**
	LOS Channel [dB]	0	Error floor [(d)]	0	**3.5**
Implementation	Technology [nm]	65	40	28	**28**
	Core Area [mm^2]	0.7	0.58	-	**2.0**
	Gate Count [kGE]	347	1,022	288	**3,607**
	Power [mW]	26.5	221	18	**127**
	Frequency [MHz]	517	425	300	**569**
	System Throughput [(e)] [Gb/s]	1.38	2.76	0.30	**1.80**
	Energy Efficiency [(f)] [pJ/b]	307	20	240	**70**
	Area Efficiency [(g)] [Mb/s/kGE]	0.24	10	0.26	**0.5**

(a): message-passing detection. (b): expectation-passing detection. (c): link margin improvement reflects to SNR gain over MMSE at BER=10^{-3}. (d): error floor occurs before BER=10^{-3} in NLOS and LOS channels. (e): system throughput assumes channel coherence within 7 OFDM symbols and K subcarriers. (f): energy efficiency is $(Power/Throughput) / (K/16)^2$. (g): area efficiency is $(Throughput/Gate\ Count) \times (K/16)^2$.

Figure 13.6.6: Comparison with state-of-the-art MIMO detector implementations.

Technology	28nm
Core Area	2mm^2
Number of BS Antennas (M)	128
Number of Users (K)	≤ 16
QAM size	4, 16, 64, 256
Channel Adaptiveness	i.i.d., NLOS, LOS
System Throughput	1.8Gb/s
Power Consumption	127mW

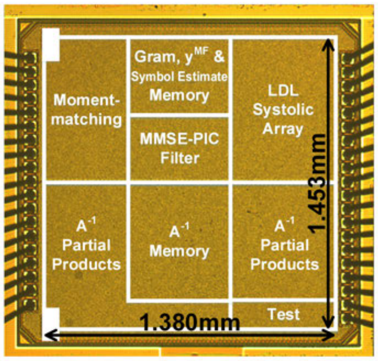

Moment-matching
Gram, y^{MF} & Symbol Estimate Memory
LDL Systolic Array
MMSE-PIC Filter
1.453mm
A^{-1} Partial Products
A^{-1} Memory
A^{-1} Partial Products
Test
1.380mm

Figure 13.6.7: Chip features and microphotograph.

978-1-5090-4940-0/18/$31.00 ©2018 IEEE

13.7 A 232-to-1996KS/s Robust Compressive-Sensing Reconstruction Engine for Real-Time Physiological Signals Monitoring

Ting-Sheng Chen, Hung-Chi Kuo, An-Yeu Wu

National Taiwan University, Taipei, Taiwan

Compressive sensing (CS) techniques enable new reduced-complexity designs for sensor nodes and help reduce overall transmission power in wireless sensor network [1-2]. Prior CS reconstruction chip designs have been described in [3-4]. However, for real-time monitoring of physiological signals, the applied orthogonal matching pursuit (OMP) algorithms they incorporate are sensitive to measurement noise interference and suffer from a slow convergence rate. This paper presents a new CS reconstruction engine fabricated in 40nm CMOS with following features: 1) A sparsity-estimation framework to suppress measurement noise interference at sensing nodes, achieving at least 8dB signal-to-noise ratio (SNR) gain under the same success rate for robust reconstruction. 2) A new flexible indices-updating VLSI architecture, inspired by the gradient descent method [5], that can support arbitrary signal dimension, (L_{new}, M), of CS reconstruction with high sparsity level (K_{max}). 3) Parallel-searching, indices-bypassing, and functional blocks that automatically group processing elements (PEs) are designed to reduce the total CS reconstruction cycle latency by 84%. Compared with prior state-of-the-art designs, this CS reconstruction engine can achieve 10× higher throughput rate and 4.2× better energy efficiency at the minimum-energy point (MEP).

In blind reconstruction algorithms that can be operated without a priori knowledge of the sparsity level, such as OMP [3-4] and stochastic gradient pursuit (SGP) [5], measurement noise destroys the sparsity (K) of received signals, degrading the reconstruction quality and speed [5]. The subspace pursuit (SP) algorithm offers excellent recovery quality and convergence under noisy scenarios. However, it is classified as *non-blind* reconstruction as it needs an *a priori* explicit sparsity level. We propose a two-phase sparsity-estimation subspace pursuit (SE-SP) CS reconstruction algorithm as shown in Figure 13.7.1. The new SE-SP can cope with measurement noise through two phases. Phase-I (P1) performs blind reconstruction similar to OMP. It reaches the maximum chosen indices numbers, K_{max}, in order to obtain all potential indices. Then, it estimates the effective sparsity level, $\hat{K}$, according to the number of elements whose amplitudes are larger than a certain noise distortion level derived from the residual norm. Phase-II (P2) applies SP with the output of P1 and the estimated $\hat{K}$ to obtain the $\hat{K}$-best sparse solution. The SE-SP still performs blind reconstruction like OMP, but it possesses all of the advantages of the non-blind SP, such as the robustness to measurement noise. Hence, it can achieve 8dB gain (time-sparse signals) in terms of *success rate* with only 10% iteration-count overhead and without any *a priori* sparsity information.

Figure 13.7.2 shows a least-mean-squares (LMS)-based architecture to implement the SP algorithm in our design. It benefits from local data updating, thus is free of global communication overhead and wiring costs. To implement the SP algorithm, we need to add/estimate $\hat{K}$ supports through a least squares (LS) computation in each iteration. However, a direct implementation of LS is unable to simultaneously realize updating of multiple indices and amplitudes, and the configurability features in [4] for handling variable measurement dimensions (M). Furthermore, it requires backward and forward substitution (BS, FS) operations, which results in additional global communication cost and high iteration counts. Although SGP uses LMS to enhance OMP, cache overhead and fixed step-size limit its scalability and convergence. This chip uses a global buffer to transpose the columns of chosen indices into an on-chip cache for the LMS updating process, resulting in 5× area reduction of the cache. It approximates the target sparse solution ($\hat{x}$) with the following advantages: 1) Speed up of the SP algorithm: it can add arbitrary support, $L_{new} \leq 128$, at each iteration; then, the sorting engine finds K-best solution for enhanced reconstruction quality. 2) Support of reconfigurable design: the line buffer-based feature of LMS is adjustable to arbitrary size (e.g., signal dimension of M and K), reaching 100% configurability with only 0.5% area overhead. 3) Local LMS updating enabling scalable designs: it relieves the limitation of global BS/FS operations. Hence, a larger signal sparsity level (K=256) and 3× higher clock rates can be achieved in this chip.

Figure 13.7.3 shows the block diagram of overall chip architecture. To achieve higher area efficiency, the tasks of the SE-SP are mapped into a folded architecture. 768KB of on-chip memory stores multiple sensing matrices for flexible reconstruction, which can be either single matrix for large or 4 matrices for small signal dimension. The 256 configurable PEs, 192KB cache and multi-task buffers can reconstruct a sparse signal with 100% flexibility. A sorting engine enables the task of finding the K-best indices/support in Phase-II. The data representation is 32b fixed point. It supports any integer of (N, M, K) up to (2048, 512, 256), representing a larger sparsity level than prior art.

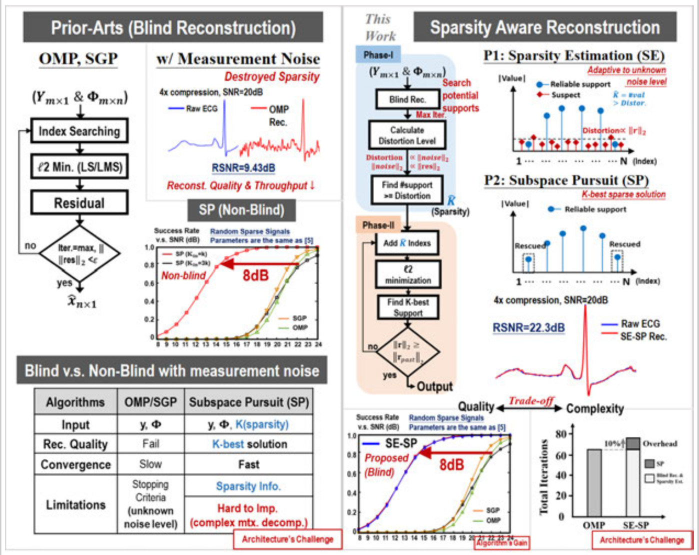
Figure 13.7.4 shows three architectural optimization techniques to reduce operating cycle count. 1) Dynamic PE grouping: because the task of index searching (IS) features high complexity but low data-dependency, the architecture can be either unfolded or folded according to the measurement size. With larger M, the PEs use multiple cycles to complete a correlation operation in IS. When M is small, the PEs are grouped dynamically to reduce the total cycle count of IS by 25% to 50%. 2) Chosen-indices bypassing: the chosen indices from past iterations can be bypassed during IS. The sorting engine checks the chosen indices before loading the column from the sensing matrix, eliminating unnecessary correlation operations and reducing cycle count by 10% when performing IS. 3) Parallel-sparsity estimation: P1 provides a sparsity-order estimation for SP to screen support, rather than reconstructing signals directly. Therefore, this chip accelerates the P1 operation by choosing multiple (2-8) indices in each iteration, which helps to reduce the total iterations by 79%. The multi-task buffer is also designed for transposing up to 8 chosen columns directly. The above optimization can effectively reduce 85% of total cycles, thus enhancing throughput rate and energy efficiency by 6.3× for CS reconstruction.

Figure 13.7.5 shows chip measurement results. The measured MEP is at 40MHz under 0.65V supply. Inspired by [4], we found that energy efficiency is nearby to the MEP. Therefore, this chip reduces area costs by using a global V_{DD} for both logic and memory. Since the SE-SP possesses a noise-tolerance feature, when reconstructing physiological signals (ECG, EMG, EEG and PPG in our measurements) under noisy conditions, it realizes at least 8dB higher SNR gain than OMP-based designs, under the same reconstruction SNR (RSNR).

Figure 13.7.6 shows a comparison with state-of-the-art designs. This CS reconstruction engine can provide 232-1996KS/s for reconstructing physiological signals of multiple patients, while offering full reconfigurability with 100% flexibility to support arbitrary signal dimensions (M, N), and robustness to measurement noise interference. By operating at a higher clock rate, but with fewer cycles, this chip achieves 7-19× throughput enhancement and 3-7× higher energy efficiency compared with prior work. The power consumption is larger than prior art due to the 3× higher operating frequency. The radar chart shows that this chip supports larger sparsity level, better energy efficiency and a higher throughput rate. Figure 13.7.7 shows the micrograph and chip summary. In conclusion, the 3.06mm^2 CS reconstruction engine can provide timely physiological signal reconstruction for data collected from CS-based wireless biosensors under noisy conditions, making intelligent patient monitoring a reality.

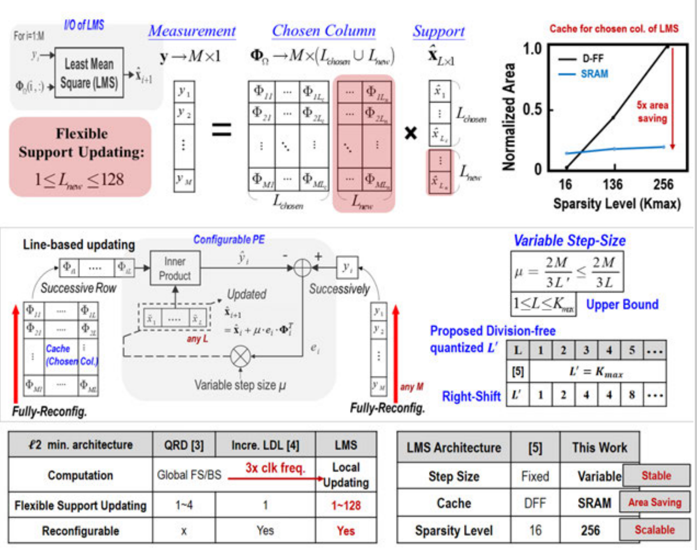
Acknowledgements:
Thank to Prof. Chia-Hsiang Yang and Prof. Tsung-Te Liu for useful discussions. The authors would like to thank National Chip Implementation Center, Taiwan for support on chip fabrication and measurements. This work is supported by Ministry of Science and Technology of Taiwan under Grant MOST 106-2221-E-002-204-MY3. The first two authors contributed equally.

References:
[1] A. Dixon, et al., “Compressed Sensing System Considerations for ECG and EMG Wireless Biosensors,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 6, no. 2, pp. 156-166, 2012.
[2] D. Gangopadhyay, et al., “Compressed sensing analog front-end for bio-sensor applications”, *IEEE JSSC,* vol. 49, no. 2, pp. 426-438, 2014.
[3] Y. -C. Cheng, et al., “Matrix-Inversion-Free Compressed Sensing with Variable Orthogonal Multi-Matching Pursuit Based on Prior Information for ECG Signals,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 10, no. 4, pp. 864-873, 2016.
[4] F. Ren, et al., “A configurable 12-to-237KS/s 12.8mW sparse-approximation engine for mobile ExG data aggregation,” *ISSCC*, pp. 334-335, 2015.
[5] Y. M. Lin, et al., “Low-Complexity Stochastic Gradient Pursuit Algorithm and Architecture for Robust Compressive Sensing Reconstruction,” *IEEE Trans. on Signal Process.*, vol. 65, no. 3, pp. 638-650, 2017.



Algorithms	OMP/SGP	Subspace Pursuit (SP)
Input	y, Φ	y, Φ, K(sparsity)
Rec. Quality	Fail	K-best solution
Convergence	Slow	Fast
Limitations	Stopping Criteria (unknown noise level)	Sparsity Info. Hard to Imp. (complex mtx. decomp.)

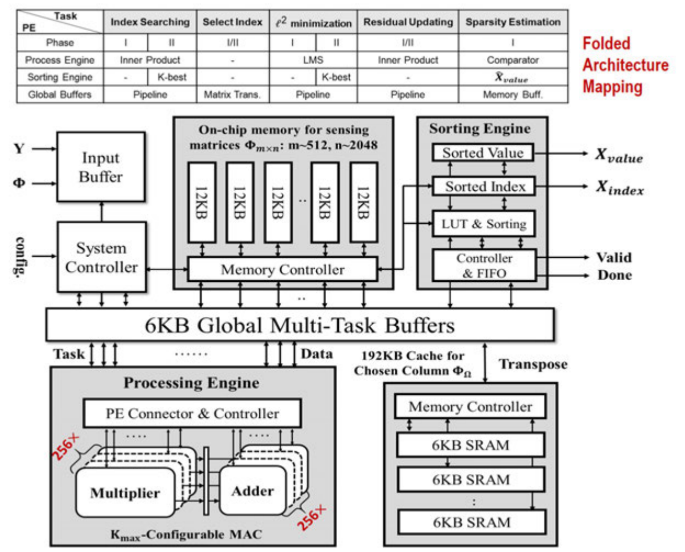
Figure 13.7.1: Proposed two-phase sparsity-aware reconstruction.



ℓ2 min. architecture	QRD [3]	Incre. LDL [4]	LMS
Computation	Global FS/BS	3x clk freq.	Local Updating
Flexible Support Updating	1~4	1	1~128
Reconfigurable	x	Yes	Yes

LMS Architecture	[5]	This Work	
Step Size	Fixed	Variable	Stable
Cache	DFF	SRAM	Area Saving
Sparsity Level	16	256	Scalable

Figure 13.7.2: LMS architecture for implementing ℓ2 minimization.



PE \ Task	Index Searching		Select Index	ℓ2 minimization		Residual Updating	Sparsity Estimation
Phase	I	II	I/II	I	II	I/II	I
Process Engine	Inner Product		-	LMS		Inner Product	Comparator
Sorting Engine	-	K-best	-		K-best	-	$\hat{x}_{value}$
Global Buffers	Pipeline		Matrix Trans.	Pipeline		Pipeline	Memory Buff.

Figure 13.7.3: Block diagram of chip architecture.

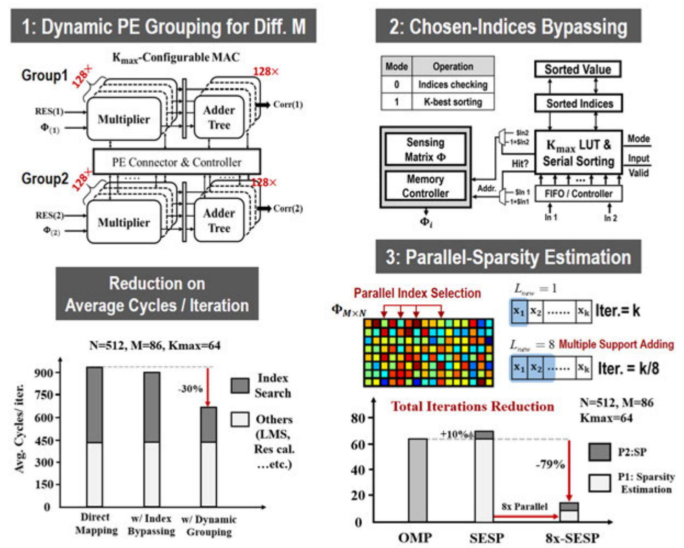


Figure 13.7.4: Architecture-level optimization.

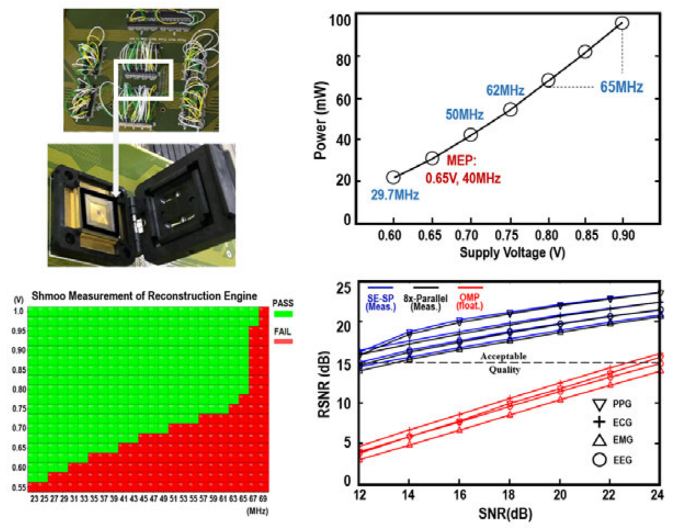
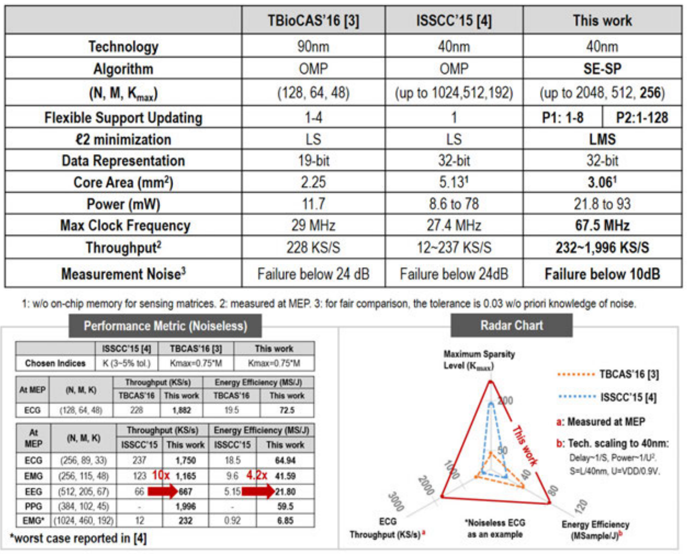


Figure 13.7.5: Measured results: Shmoo measurement, power consumptions, and reconstruction quality vs. SNR.



	TBioCAS'16 [3]	ISSCC'15 [4]	This work	
Technology	90nm	40nm	40nm	
Algorithm	OMP	OMP	**SE-SP**	
(N, M, K_{max})	(128, 64, 48)	(up to 1024,512,192)	(up to 2048, 512, **256**)	
Flexible Support Updating	1-4	1	**P1: 1-8**	**P2:1-128**
ℓ2 minimization	LS	LS	**LMS**	
Data Representation	19-bit	32-bit	32-bit	
Core Area (mm^2)	2.25	5.13[1]	**3.06**[1]	
Power (mW)	11.7	8.6 to 78	21.8 to 93	
Max Clock Frequency	29 MHz	27.4 MHz	**67.5 MHz**	
Throughput[2]	228 KS/S	12~237 KS/S	**232~1,996 KS/S**	
Measurement Noise[3]	Failure below 24 dB	Failure below 24dB	**Failure below 10dB**	

1: w/o on-chip memory for sensing matrices. 2: measured at MEP. 3: for fair comparison, the tolerance is 0.03 w/o priori knowledge of noise.

Performance Metric (Noiseless)

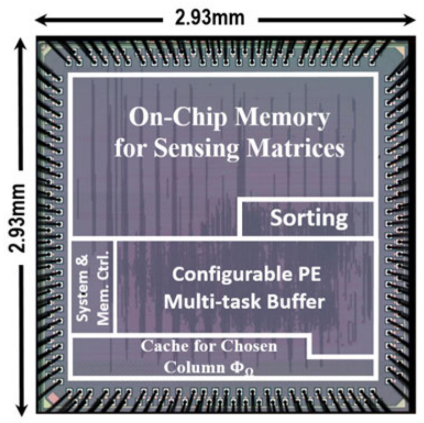
	ISSCC'15 [4]	TBCAS'16 [3]	This work
Chosen Indices	K (3~5% tol.)	Kmax=0.75*M	Kmax=0.75*M

At MEP	(N, M, K)	Throughput (KS/s)		Energy Efficiency (MS/J)	
		TBCAS'16	This work	TBCAS'16	This work
ECG	(128, 64, 48)	228	**1,882**	19.5	**72.5**

At MEP	(N, M, K)	Throughput (KS/s)		Energy Efficiency (MS/J)	
		ISSCC'15	This work	ISSCC'15	This work
ECG	(256, 89, 33)	237	**1,750**	18.5	**64.94**
EMG	(256, 115, 48)	123	10x **1,165**	9.6	4.2x **41.59**
EEG	(512, 205, 67)	66	**667**	5.15	**21.80**
PPG	(384, 102, 45)	-	**1,996**	-	**59.5**
EMG*	(1024, 460, 192)	12	**232**	0.92	**6.85**

*worst case reported in [4]

Figure 13.7.6: Comparison with prior chip implementations.


2.93mm
2.93mm
On-Chip Memory for Sensing Matrices
Sorting
System & Mem. Ctrl.
Configurable PE Multi-task Buffer
Cache for Chosen Column Φ_Ω

Technology	40nm 1P10M CMOS
Chip Size	2.93 × 2.93 mm^2
Core Size*	3.06 mm^2
Core V_{DD}	0.9V
I/O V_{DD}	2.5V
Clock Freq.	67.5 MHz
Power	93mW @0.9V

* w/o on-chip memory for sensing matrices
Measured at room temperature

Figure 13.7.7: Chip micrograph and summary.