

37.3 Monolithic In-Memory Computing Microprocessor for End-to-End DNN Inference in MRAM-Embedded 28nm CMOS Technology with 1.1Mb Weight Storage

Soonwan Kwon^{1,2}, Sungmeen Myung¹, Jangho An¹, Hyunsoo Kim¹, Minje Kim¹, Hyungwoo Lee¹, Wooseok Yi¹, Seungchul Jung¹, Daekun Yoon¹, Shinhee Han³, Saeyoon Chung³, Kilho Lee³, Jeong-Heon Park³, Kangho Lee³, Sang Joon Kim¹, Donhee Ham^{1,4}

¹Samsung Advanced Institute of Technology, Suwon, Korea

²Seoul National University, Seoul, Korea

³Samsung Electronics, Giheung, Korea

⁴Harvard University, Cambridge, MA

Always-on AI sensor applications—based on deep neural networks (DNNs)—with sparse inference require low power consumption during both computing and idle phases. In-memory computing (IMC) with non-volatile memory crossbar arrays could meet this demand. Concretely, the co-location of memory (DNN weight storage) and computing (analog matrix multiplications (MMs)) in crossbar arrays obviates the need to shuttle weight data, thus reducing the computing power consumption, while the use of the non-volatile memory minimizes the power consumption during idle states. Resistive, phase-change, and magneto-resistive random-access memory (RRAM, PRAM, and MRAM) and flash memory are well-known non-volatile memory types, with their own merits and drawbacks. Of these, MRAM boasts high endurance and low switching energy (with the drawback being 1b storage per cell) [1] and has been commercially embedded in CMOS logic technology, and thus MRAM is a good non-volatile memory candidate for IMC. However, previous MRAM IMC works [2-5] have been focused on individual crossbar arrays, which correspond only to a small fraction of a DNN.

We realize a monolithic IMC microprocessor capable of end-to-end DNN computing in MRAM-embedded 28nm CMOS logic technology. It fully integrates a mixed-signal data path consisting of many MRAM crossbar arrays for analog MMs and digital logic for non-MM processing (such as pooling and activation) to map all DNN layers (Fig. 37.3.1). Specifically, the datapath features 14 analog MM engines—comprising 126 MRAM crossbar arrays and 1,008 transimpedance amplifier (TIA) and analog-to-digital converter (ADC) pairs—for analog MMs and their digital conversion, and 1 post engine (digital logic) for non-MM processing. Computed data is routed to and from analog and digital sectors fully on chip to handle all DNN layers, while managing analog noise.

Figure 37.3.2 illustrates the datapath backbone of our processor, consisting of the 14 MM engines and 1 post engine. Each MM engine, comprising nine 128×64 MRAM crossbar arrays and their readout electronics (8 TIA-ADC pairs per crossbar array), executes analog MMs and digital conversion. Concretely, the MM engine takes nine 1×128 input vectors and produces nine 1×64 digitized vectors, which are digitally added to one 1×64 digitized vector as the final output. Effectively, the MM engine multiplies a 1×1,152 input vector with a 1,152×64 weight matrix to yield a 1×64 digitized output vector. The post engine (digital logic) performs non-MM processing. The datapath has a total of 126 crossbar arrays with ~1Mb weight storage capacity and 1,008 TIA-ADC pairs. For computing of a DNN layer, we typically use one MM engine followed by the post engine to perform MMs and subsequent non-MM processing. Its output is then fed to the next MM engine followed by the post engine, for computing the next DNN layer. This is repeated for end-to-end DNN inference, controlled by a microcontroller unit (MCU) integrated on the same chip.

Figure 37.3.3 illustrates the dataflow for a 3×3 convolution in an MM engine, which consists of 3 processing elements (PEs), with each containing 3 crossbar arrays. These arrays share inputs and control circuits (except enable signals) to reduce routing complexity. To enhance throughput, the MM engine takes 4 lines of inputs (e.g., X_1, X_5, X_9, X_{13}) through an input fetcher unit (IFU) and loads them into a shared shift register, which aligns the inputs to efficiently feed the 3 PEs. The MM engine processes these inputs over a 4-cycle period, producing 2-line outputs in a zig-zag pattern, which are sent directly to the post engine (if pooling is performed in the post engine, the output is condensed into 1 line). Although the throughput (0.5 outputs per cycle) is still lower than that of a fully connected dataflow (1 output per cycle), the benefits of this architecture—reduced connection complexity and lower SRAM bandwidth requirements—are significant, particularly in resource-constrained always-on AI sensing applications. Furthermore, the dataflow utilizes vertical overlap between input feature map (IFM) tiles, reducing SRAM accesses required for IFM fetching by a factor of 0.67, as compared to line-by-line processing [6].

Figure 37.3.4 presents the schematic of a 128×64 MRAM crossbar array. Each bitcell, which stacks 8 identical unit structures, with each unit featuring 2 MTJs and 2 FETs, performs a binary multiplication, which is akin to the XNOR operation in the digital domain. This switching-based analog multiplication also works for a bitcell with only 1 unit structure [3], but we use 8 unit structures per bitcell to increase the bitcell resistance for relatively low-power operation. The MRAM crossbar array with 64 columns, followed by 8 TIA-ADC pairs (each TIA-ADC pair is multiplexed to 8 columns), produces 64 digitized multiply-and-accumulate (MAC) outputs (each ADC is a 4b flash ADC), thus enabling the aforementioned MMs. All crossbar arrays have two individual calibration points and share common reference voltages.

Figure 37.3.5 depicts system-wide calibrations aimed at mitigating prominent circuit non-idealities, in particular, column-to-column variations, layer-to-layer variations, and residual noise. 1) To mitigate the column-to-column variations, each column-group, consisting of 8 columns that share an TIA-ADC pair, is initially tuned over a broad range by compensating for the offset voltage of the op-amp in the TIA and finely tuned by controlling the resistance in the TIA by comparing the measured output with the target output. After completing the calibration of all the column groups within a chip, 95% of errors in 126 crossbar arrays can be reduced within ± 1 LSB. 2) Layer-to-layer dependent variations arise because the number of weights generally varies from layer to layer in a DNN, and thus different MM engines use generally different numbers of crossbar arrays, drawing different average current from the power supply. This causes layer-to-layer variations in offsets in MAC output values. These variations can be experimentally estimated by using predefined patterns of inputs and can be subsequently calibrated out (this correction incurs no additional cost because the estimated variations are combined into bias-addition parameters). This calibration of the layer-dependent variations is crucial for achieving software-comparable accuracy (without the calibration, inferring accuracy with MNIST dataset is decreased by nearly 10%). 3) Residual noise encompasses all remaining errors after calibrating both column- and layer-dependent variations. To enhance the robustness of inference against this residual noise, we do noise-aware training. ReLU activation does not work well for DNNs with many layers, due to deep back propagation of noise. The hyperbolic tangent (tanh) activation does not back propagate noise as deeply, but it suffers from the gradient vanishing problem during training. To address this, we perform noise-aware training as follows. First, we train the DNN with ReLU for all layers and with no noise. Then, we convert the activation function of the first layer to tanh, and retrain the DNN, now with noise added to the first layer. Subsequently, we convert the activation function of the second layer to tanh, and retrain the DNN, with noise added to the first and second layers, while not updating the weights of the first layer. We repeat this process until all activation functions become tanh. Ultimately, for inference, our processor uses the tanh activation for all layers.

Figure 37.3.6 presents system-level measurements. The power efficiencies of a representative individual MRAM crossbar array and a representative IMC microprocessor in its entirety are 59.8TOPS/W and 20TOPS/W, respectively. The power breakdown across all processor components was measured for 100 chips, and its average is also shown in the figure. Under the identical clock and supply voltage used for the power measurement, we classify MNIST handwritten digits using an 8-layer convolutional neural network fully mapped onto the IMC microprocessor with dataflow fully on chip, achieving an accuracy of 97.62% (software baseline: 99.45%). In addition, we further evaluated the face-detection model with the FDDB dataset, with all weights fully deployed on chip, but with external feature map tiling (due to large image), achieving an accuracy of 91.3% (software baseline: 92.51%). Figure 37.3.7 shows a die photo and the performance summary.

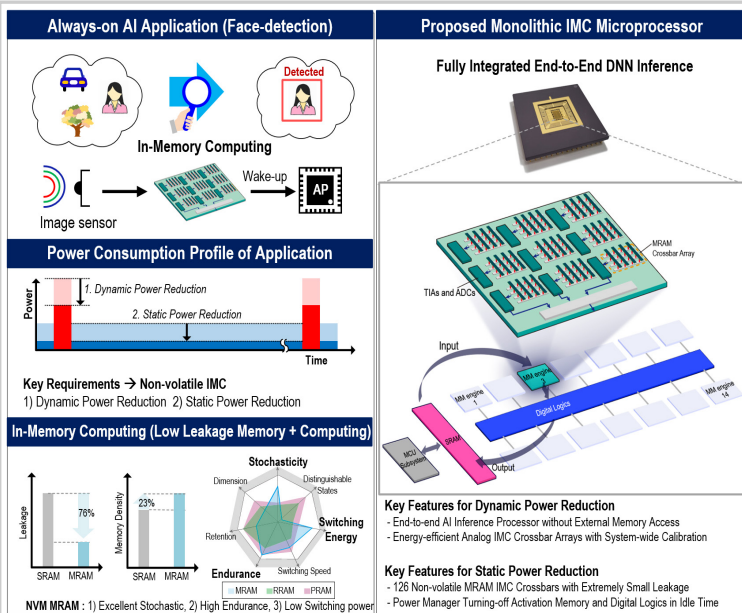


Figure 37.3.1: Edge AI system overview, motivation for end-to-end IMC microprocessor.

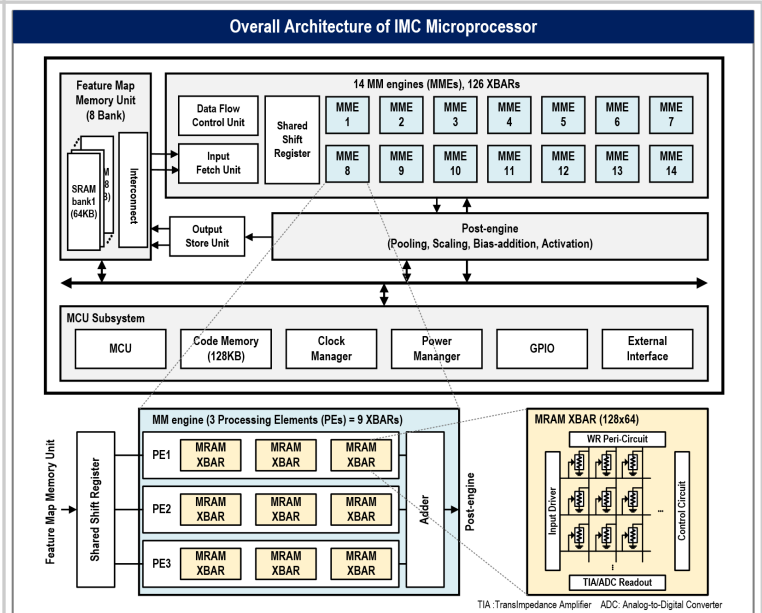


Figure 37.3.2: Overall architecture of IMC microprocessor.

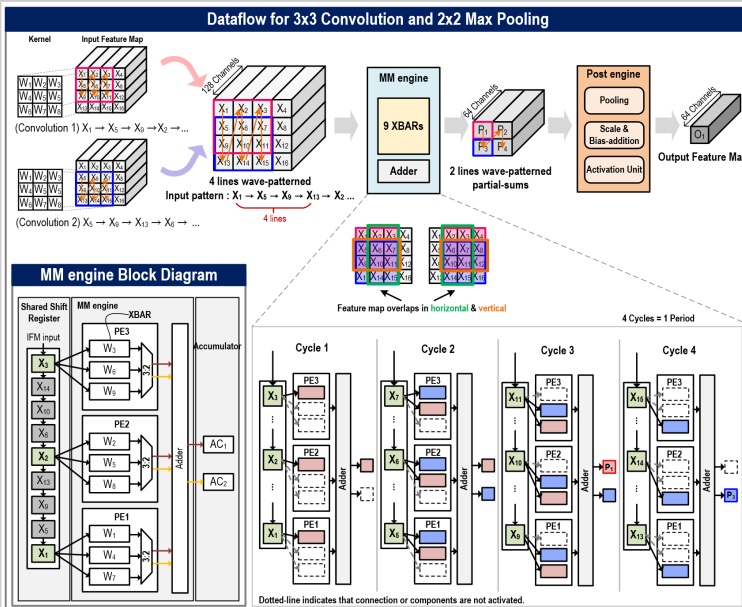


Figure 37.3.3: Dataflow: Example for 3x3 convolution with max-pooling.

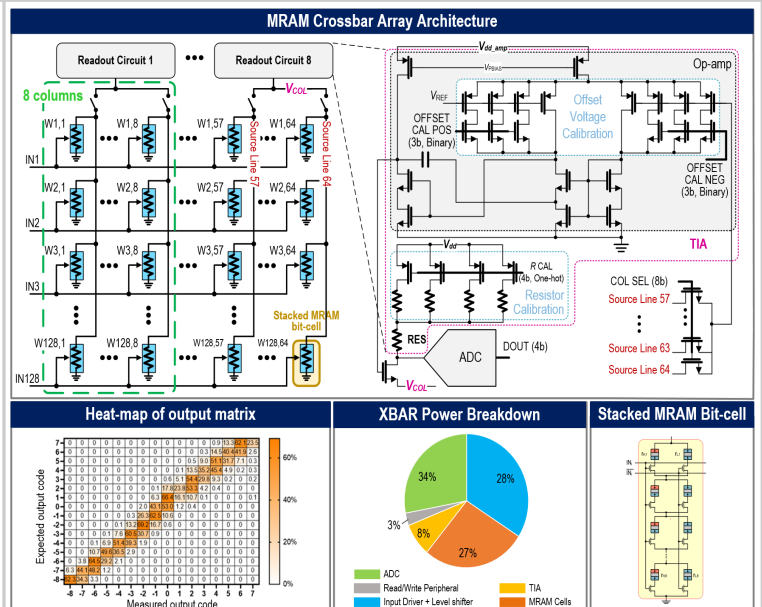


Figure 37.3.4: MRAM crossbar array and readout electronics.

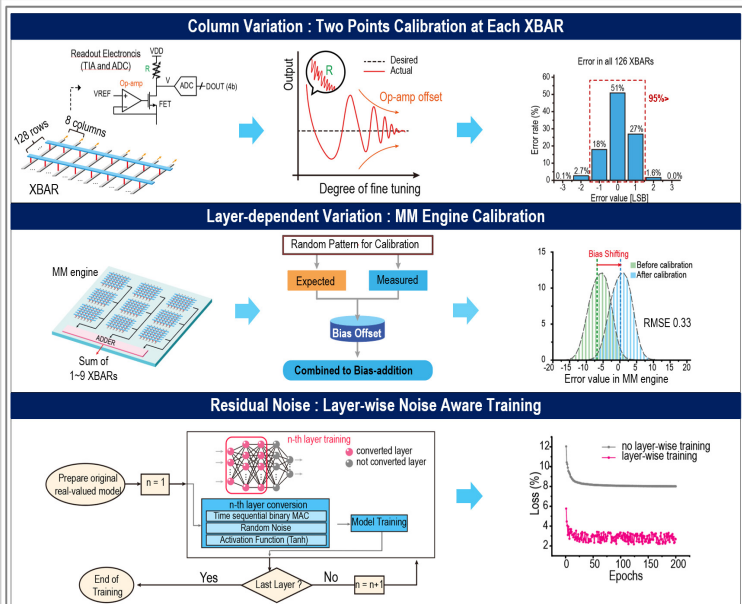


Figure 37.3.5: A system-wide calibration approach.

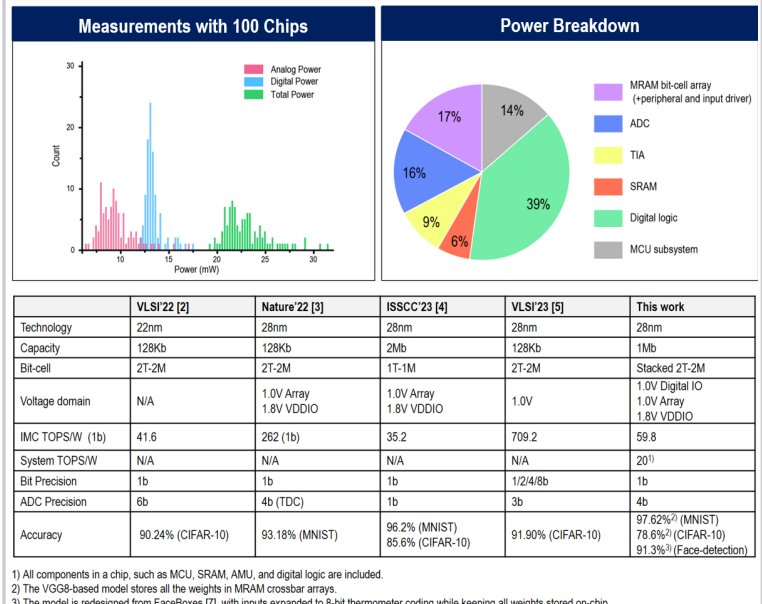


Figure 37.3.6: Comparison with prior work.

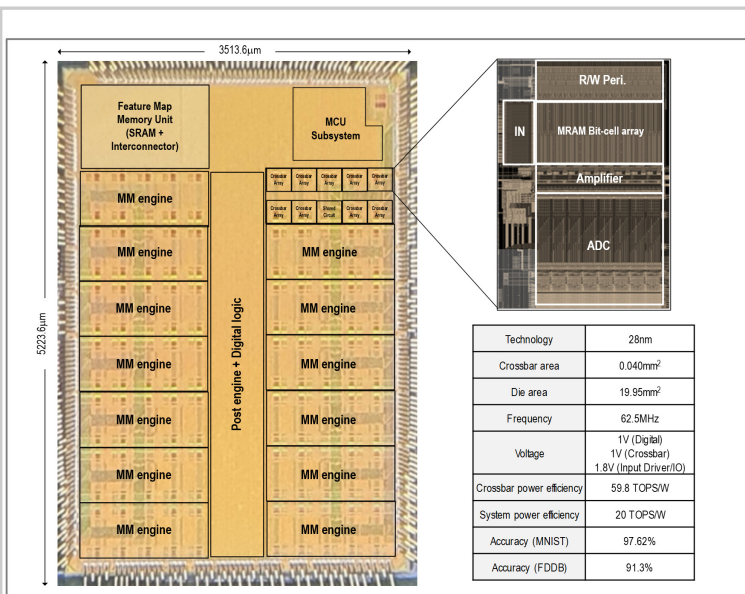


Figure 37.3.7: Die photo and performance summary.

References:

- [1] Shao, Qiming, Zhongrui Wang, and J. Joshua Yang, "Efficient AI with MRAM," *Nature Electronics*, vol. 5, pp. 67-68, 2022.
- [2] P. Deaville et al., "A 22nm 128-kb MRAM Row/Column-Parallel In-Memory Computing Macro with Memory-Resistance Boosting and Multi-Column ADC Readout," *IEEE Symp. VLSI Circuits*, pp. 268-269, 2022.
- [3] S. Jung et al., "A Crossbar Array of Magnetoresistive Memory Devices for In-Memory Computing," *Nature*, vol. 601, pp. 211-216, 2022.
- [4] H. Cai et al., "A 28nm 2Mb STT-MRAM Computing-in-Memory Macro with a Refined Bit-Cell and 22.4 - 41.5TOPS/W for AI Inference," *ISSCC*, pp. 500-502, 2023.
- [5] W. Xie et al., "A 709.3 TOPS/W Event-Driven Smart Vision SoC with High-Linearity and Reconfigurable MRAM PIM," *IEEE Symp. VLSI Circuits*, 2023.
- [6] S. Yin et al., "PIMCA: A 3.4-Mb Programmable In-Memory Computing Accelerator in 28nm for On-Chip DNN Inference," *IEEE Symp. VLSI Circuits*, 2021.
- [7] Zhang, Shifeng et al., "FaceBoxes: A CPU Real-Time Face Detector with High Accuracy," *IEEE International Joint Conf. on Biometrics*, 2017.