A 4-bit Mixed-Signal MAC Array with Swing Enhancement and Local Kernel Memory

Wei-Han Yu[1,2], Massimo Giordano[2], Rohan Doshi[2], Minglei Zhang[1], Pui-In Mak[1], Rui P. Martins[1,3], and Boris Murmann[2]

1 - The State-Key Laboratory of Analog and Mixed-Signal VLSI/IME and FST-ECE, University of Macau, Macao China, 2 - Stanford University, USA
3 - on-leave from the Instituto Superior Técnico, UL, Portugal

2021 IEEE International Midwest Symposium on Circuits and Systems (MWSCAS) | 978-1-6654-2461-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/MWSCAS47672.2021.9531769

***Abstract*—Modern deep neural networks require energy- and area-efficient multi-bit multiply-accumulate (MAC) functions. In-memory computing (IMC) with analog accumulation has shown the potential to outperform purely digital solutions but lacks efficient multi-bit computation. In this work, we explore the design of a mixed-signal, charge-sharing compute array that performs 4-bit MAC operations within one clock cycle. Its key features include 7×4 bit kernel memory in each MAC cell, as well as a per-kernel binary combiner that simplifies the unit cell design and enables efficient in-column ADC integration. In addition, it leverages a differential switching scheme that improves the signal swing by 4× relative to single-ended schemes, thereby reducing ADC energy. Post-layout simulations in 28 nm CMOS indicate an energy efficiency of 6.4 fJ/MAC and a compute density of 3.17 TOPS/mm^2 for an input activation vector size of 160.**

Keywords— Deep neural networks, hardware accelerators, in-memory computing, mixed-signal integrated circuits, switched capacitor circuits.

I. Introduction

Machine learning inference using modern deep neural networks (DNNs) can be accelerated using domain-specific hardware and massively parallel matrix-vector multiplication. While state-of-the-art digital implementations leverage this opportunity to demonstrate impressive energy savings [1], even larger gains may be possible by invoking mixed-signal compute fabrics [2]. Specifically, dense analog accumulation of partial products can help eliminate and/or better amortize buffer memory accesses that amount to a significant fraction of the total processing energy. A variety of mixed-signal multiply-accumulate (MAC) arrays [3-6] have been developed to explore this potential. Among them are designs based on in-memory computing (IMC), which typically aim to maximize area density at the expense of supporting only single-bit computations in each unit cell. As shown in [5], single-bit IMC cores can be used to perform multi-bit operations sequentially, but this leads to diminishing returns due to multiple A/D conversions and array access operations.

This work aims to strike a middle ground between highly restricted IMC unit elements and the relatively large digital processing elements that lie at the other end of the spectrum. We study the efficacy of mixed-signal MAC elements that perform 4-bit multiplications in a single cycle through capacitive charge sharing. The choice of 4 bits is linked to the finding that a pareto-optimal tradeoff between inference accuracy and model size is typically achieved for bitwidths in the range of 2-4 [7]. Our design is based on the concepts described in [2] and features a swing-enhancement approach as well as local kernel memory that can be placed underneath the compute-capacitors in higher metal layers. Simulation results in 28 nm CMOS indicate competitive compute energy (6.4 fJ/MAC) and density (3.17 TOPS/mm^2) for an input vector dimension of 160.

This work was supported in part supported by the Macao Science and Technology Development Fund (FDCT), Macao SAR (File no. File no. 0110/2019/A2 and SKL-AMSV(UM)-2020-2022), Stanford's SystemX Alliance, Semiconductor Research Corporation Task No. 2810.078, and National Science Foundation Grant No. 1937294.

II. Circuit Implementation

A. Architecture

Fig. 1 shows a conceptual schematic of the $M\times N$ MAC array. Each of the M Kernels contain N-dimensional dot-product circuitry with input activations d and weights w. The input activations are broadcast across the array to amortize memory access and to yield a full matrix-vector product. Each of the N MAC elements within a kernel performs a 4-bit multiplication with subsequent capacitive accumulation.

The multiplications are performed similar to a digital multiplier, which uses bitwise AND operations and a digital adder network to collect the partial products. Our mixed-signal design keeps the AND operations in the digital domain but sums the partial products through analog charge sharing on six analog wires (V_{MAC}). Each V_{MAC} wire has a different binary weight and the product bit cells are connected accordingly (e.g., $d_0\times w_0$ to $V_{MAC,0}$). The weight bit w_3 is a sign bit that flips the accumulation polarity. The input activations are unsigned, corresponding to rectified linear unit (ReLU) outputs.

The wire voltages are capacitively summed in the shown binary combiner block before digitization with an 8-bit SAR ADC (one per kernel). An alternative architecture would use a binary-weighted capacitor array to perform the weighted charge sharing locally (within each MAC element) and without using a per-kernel combiner. However, this approach requires 105 unit-capacitors in each MAC element, which is prohibitively large and complex. The final dot product result for each kernel is formed by charge sharing between all N MAC elements using the same V_{MAC} wires. Thus, the analog charge sharing eliminates all digital partial product additions that would normally occur within multipliers and in the accumulator hardware.

The weight bits are stored in a custom-designed local 6T SRAM inside each MAC element (see Fig. 2). The memory size per MAC element can be chosen based on the application and the dimensions of the array. In this work, we use a configuration with seven 4-bit weights per MAC element. Having dense memory right next to the product cells allows us to load the

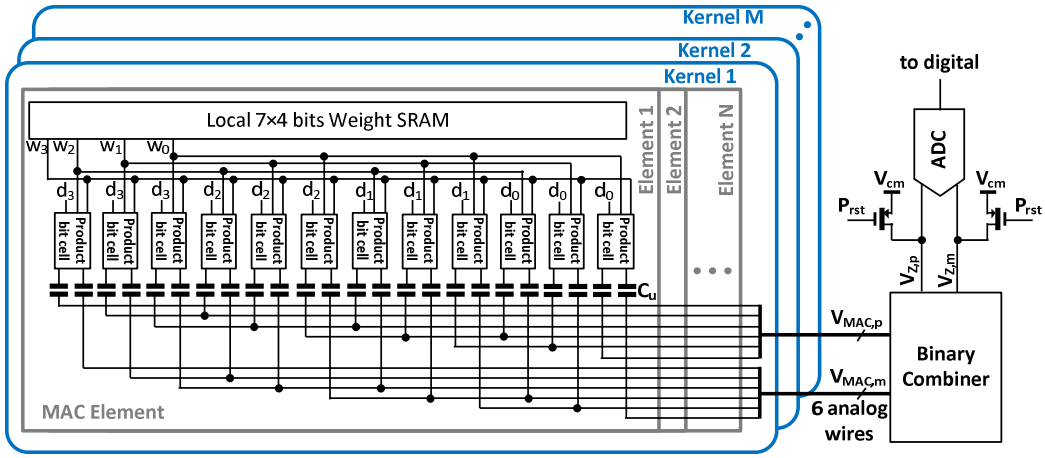


Fig. 1. Mixed-signal MAC array with binary combiner and local memory.

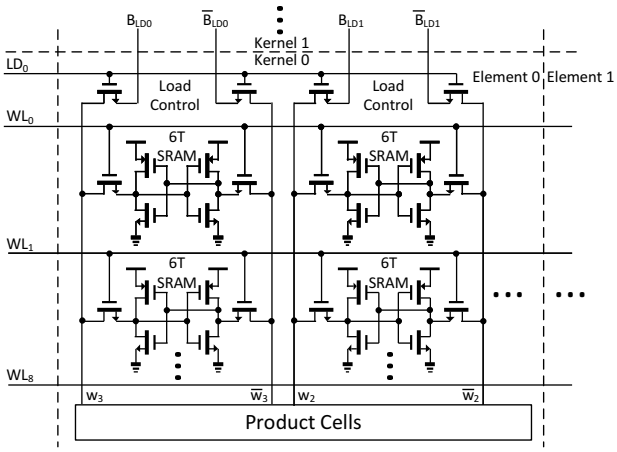
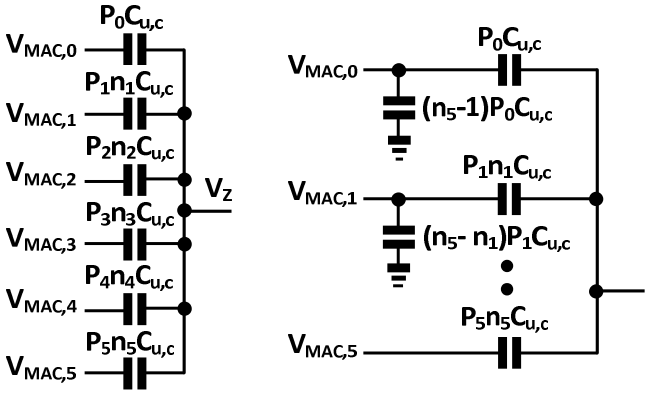


Fig. 2. 6T SRAM for local weight storage.



(a) (b)

Fig. 3. Binary combiner with (a) series compensation, and (b) parallel compensation.

weights in a single clock cycle, reducing read energy and data movement. This increased read bandwidth avoids idling the compute when loading new weights in DNN layers that have small input dimensions but a large number of weights. Furthermore, given the small array size, the SRAM cells can directly drive the MAC logic, and energy-hungry sense amplifiers can be avoided. Each word line (WL) and load data (LD) signal is shared across all MAC elements, while the bit lines for data load (B_{LD}) are shared across kernels. Thus, the number of control signals scales with the local memory and array dimensions (N+M), as opposed to the number of MAC elements (M×N).

B. Binary Combiner

The binary combiner consists of weighted capacitors, as shown in Fig. 3(a). The combined output can be determined through a nodal analysis at V_z:

$$V_z = \sum_{k=0}^{5} \frac{V_{MAC,k} P_k N C_u n_k P_k C_{u,c}}{N P_k C_u + n_k P_k C_{u,c}} \cdot \sum_{k=0}^{5} \frac{N P_k C_u + n_k P_k C_{u,c}}{N P_k C_u n_k P_k C_{u,c}} \quad (1)$$

Here, k is the wire index, $C_{u,c}$ is the unit capacitance in the binary combiner, $V_{MAC,k}$ is the V_{DD}-normalized dot product, n_k are the ratios between the (nominally) binary combiner capacitors, and P_k is a ratio defined by the number of product cells connected to the analog wire within a MAC element. For a 4-bit MAC, P_k is [1, 2, 3, 3, 2, 1]. The second term in (1) is independent of V_{MAC} and normalizes V_z to V_{DD}. By inspecting the first quotient in (1), we see that the denominator term $n_k P_k C_{u,c}$ introduces nonlinearity, caused by voltage division between the array and combiner capacitances. To overcome this issue, we consider series compensation and parallel compensation (see Fig. 3(b)).

Series compensation alters the ratios n_k to restore the binary weighting. This is achieved when adjacent terms inside the first summation of (1) ($k = n$ and $w = k$-1) have a ratio of 2:

$$\frac{N C_u n_k C_{u,c}}{N C_u + n_k C_{u,c}} \Big/ \frac{N C_u n_{k-1} C_{u,c}}{N C_u + n_{k-1} C_{u,c}} = 2 \quad (2)$$

Solving (2), we find:

$$n_k = \frac{2 N C_u n_{k-1}}{N C_u - n_{k-1} C_{u,c}} \quad (3)$$

Note that the required n_k depend on the array dimension N, which is a disadvantage of this approach. On the other hand,

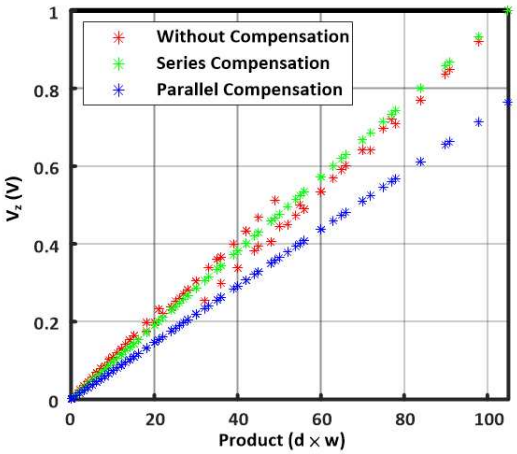


Fig. 4. Simulated linearity with and without combiner compensation (N = 32 and $C_u/C_{u,c}$ = 2).

unlike parallel compensation, series compensation incurs no attenuation. Thus, a larger $C_{u,c}$ can be chosen for better capacitance matching and reduced attenuation from the ADC's input capacitance.

For parallel compensation, grounded capacitors are added before the combiner to ensure that all wires have the same load capacitance. Applying a T to π transformation for the capacitors and performing a nodal analysis at V_z gives:

$$V_z = \frac{\sum_{k=0}^{5} V_{MAC,k} P_k N C_u n_k P_k C_{u,c}}{\sum_{k=0}^{5} \left[P_k N C_u n_k P_k C_{u,c} + n_k (n_5 - n_k) (P_k C_{u,c})^2 \right]} \quad (4)$$

The second part of the denominator stems from this scheme's capacitive attenuation. The attenuation ratio can be written as:

$$V_z = \frac{N C_u}{N C_u + \sum_{k=0}^{5} n_k (n_5 - n_k) C_{u,c}} \quad (5)$$

The attenuation gets smaller as C_u and N get larger. Yet, smaller $C_{u,c}$ leads to higher attenuation from the ADC's input capacitance. Compared to series compensation, capacitance matching for parallel compensation is easier to achieve since all compensation capacitors are integer multiples of $C_{u,c}$, independent of N.

Fig. 4 shows the simulated circuit linearity with and without compensation for N = 32, and a C_u-to-$C_{u,c}$ ratio of 2. For series compensation, the n_k values computed using (3) are [1 2.03 4.20 8.98 20.90 62.06] instead of 2^k. For parallel compensation, the simulated attenuation factor of 0.76 matches the analysis.

C. Split-Capacitor Product Bit Cell

Fig. 5(a) shows the implementation of the product bit cells. It employs a split unit capacitor configuration to enhance the MAC array's differential output swing while maintaining a fixed common-mode voltage for the ADC. The swing increment is beneficial since the variance of V_Z scales inversely proportional with N [2], and is only a very small fraction of V_{DD} for large arrays. This translates into a reduced ADC full-scale range and a higher conversion energy to overcome thermal noise. Assuming that the circuit is thermal noise limited, a 2× reduction in full-scale range quadruples the ADC energy. From

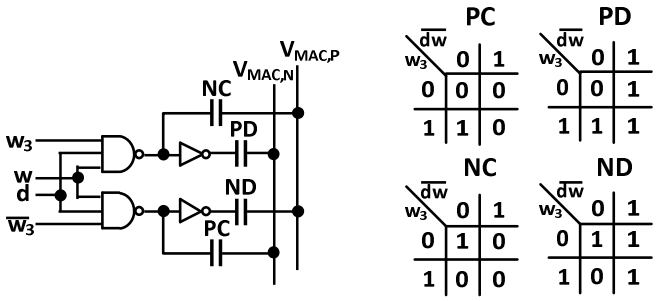


Fig. 5. (a) Schematic and (b) logic table for the split-capacitor MAC-bit cell.

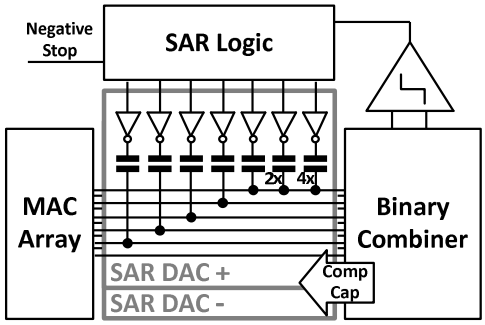


Fig. 6. MAC-integrated SAR ADC.

simulation, the ADC energy is comparable to the array energy for 4-bit MAC cells and N = 64.

The bit cell's operation is summarized in Fig. 5(b). Before the MAC operation, d_3-d_0 are reset to 0 (all $\overline{dw}$=1), and V_z is initialized to V_{cm} to define the common-mode for the ADC. During operation, when the bit product $\overline{dw}$ = 0, depending on the sign bit (w_3), $V_{MAC,p}$ is either charged (w_3=1) or discharged (w_3=0). $V_{MAC,m}$ produces the opposite swing to maintain the same common-mode voltage.

Compared to the single-ended two's complement charge sharing, the split-capacitor product bit cell computes the one's complement product on the differential wires, boosting the voltage by 4×. This allows for a significant reduction in ADC energy. Moreover, unlike [3] and [4], which require differential d inputs, our scheme uses a single-ended d that aids to reduce the wire energy. Minimum sized logic gates and HVT devices are used to minimize the computation energy. The higher latency from HVT devices is acceptable since the array speed is limited by the ADC instead of the MAC core.

D. Array-integrated SAR ADC

As shown in Fig. 6, the SAR ADC's DAC is integrated into the MAC array before the binary combiner to avoid extra binary-weighted capacitors for the SAR DAC. Conceptually, this scheme is similar to the prior work of [8, 9] with column-integrated DACs. Moreover, the parallel compensation capacitors can be absorbed into the DAC, utilizing the attenuation for SAR ADC binary searching. The capacitors of the two MSBs can be rerouted for maximum utilization of the attenuation range, and the remaining range can be used for DNN biases and comparator offset calibration. The SAR ADC features a programable negative stop control. If no post-kernel

TABLE I. PERFORMANCE COMPARISON.

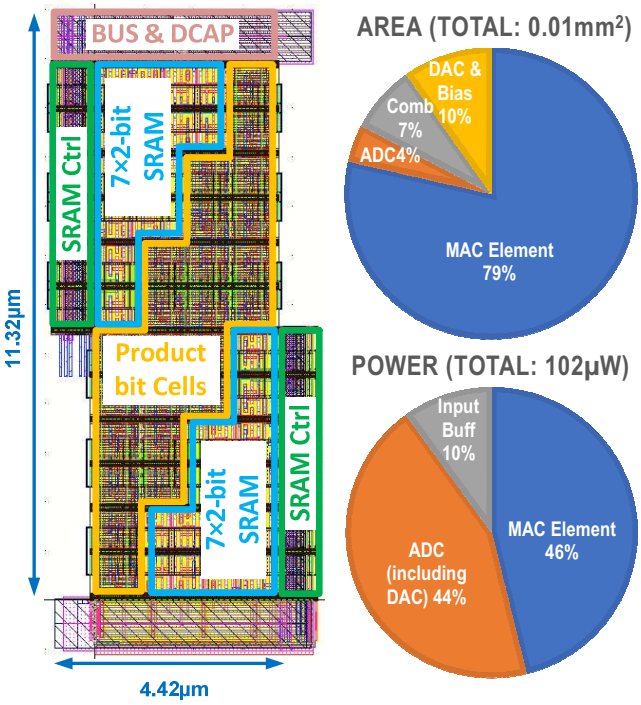
	This Work (Simulated)	Y. -D. Chih ISSCC'21 [1]	H. Jia ISSCC'21 [5]
Technology	28 nm CMOS	22 nm CMOS	16 nm CMOS
4-bit MAC Throughput (MMAC/s)	100	100	50
4-bit MAC Elements per Column	160	256	288
Accumulation Domain	Analog	Digital	Analog, In-memory
In-Element Memory (bits)	7×4	4	4
Precision (d, w)	4, 4	4, 4	4, 4
Energy Efficiency (fJ/MAC)	6.4	22.5	16.5
Area Efficiency (TOPS/mm²)	3.17	16.3	2.67

accumulation is needed and a ReLU output is assumed, the ADC can be stopped after the MSB decision to save energy.

III. SIMULATION RESULTS AND CONCLUSIONS

A prototype of a parallel-compensated MAC kernel is built in 28 nm CMOS with a 0.9 V supply. The layout of one MAC element is shown in Fig. 7 (a). C_u is built directly on top of the product bit cells using two of the higher metal layers to minimize the parasitics. The product bit cells are grouped on different rows according to their accumulation wire to reduce the parasitic capacitance. We performed post-layout simulation for a MAC array with N = 160, operating at 100 MHz with an input activation activity factor of 0.5 and static weights. The area and power breakdowns are depicted in Fig. 7 (b). The combined RMS noise is <350 µV assuring <1% top-1 accuracy degradation (i.e., ~39 dB SNR) for popular DNNs deployed on the ImageNet dataset [11]. We also carried out mismatch and nonlinearity simulations, which show that a C_u of 475 aF [10] is sufficient to maintain <1% top-1 accuracy degradation.

Comparing with the prior art, this work offers 4-bit MAC operations with a single ADC conversion, resulting in a higher energy efficiency than single-bit IMC configured for 4-bit precision [5]. This work also introduces a 7×4-bit in-element memory that can help reduce DNN data movement. Compared to a recent digital implementation [1], 3.5x higher energy efficiency is achieved.



(a) (b)

Fig. 7. (a) Layout of the MAC element in 28 nm CMOS and (b) power

REFERENCES

[1] Y. -D. Chih et al., "An 89TOPS/W and 16.3TOPS/mm2 All-Digital SRAM-Based Full-Precision Compute-In Memory Macro in 22nm for Machine-Learning Edge Applications," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 252–253.

[2] B. Murmann, "Mixed-Signal Computing for Deep Neural Network Inference," *IEEE Tran. Very Large Scale Integ. (VLSI) Systems*, 2020.

[3] D. Bankman, L. Yang, B. Moons, M. Verhelst, and B. Murmann, "An Always-On 3.8 J/86% CIFAR-10 Mixed-Signal Binary CNN Processor with All Memory on Chip in 28-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 54, no. 1, pp. 158–172, Jan. 2019.

[4] Z. Jiang, S. Yin, J. Seo, and M. Seok, "C3SRAM: An In-Memory-Computing SRAM Macro Based on Robust Capacitive Coupling Computing Mechanism," *IEEE J. Solid-State Circuits*, vol. 55, no. 7, pp. 1888-1897, Jul. 2020.

[5] H. Jia et al., "A Programmable Neural-Network Inference Accelerator Based on Scalable In-Memory Computing," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 236–237.

[6] H. Valavi, P. J. Ramadge, E. Nestler, and N. Verma, "A 64-Tile 2.4-Mb In-Memory-Computing CNN Accelerator Employing Charge-Domain Compute," *IEEE J. Solid-State Circuits*, vol. 54, no. 6, pp. 1789–1799, Jun. 2019.

[7] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, "Learned Step Size Quantization," *arXiv*: 1902.08153, Feb. 2019.

[8] D. Bankman, J. Messner, A. Gural and B. Murmann, "RRAM-Based In-Memory Computing for Embedded Deep Neural Networks," *Proc. Asilomar Conference*, Nov. 2019, pp. 1511-1515.

[9] C. Yu, T. Yoo, T. Kim, K. Chai, and B. Kim, "A 16K Current-Based 8T SRAM Compute-In-Memory Macro with Decoupled Read/Write and 1-5bit Column ADC," *IEEE CICC*, Mar. 2020.

[10] V. Tripathi and B. Murmann, "Mismatch Characterization of Small Metal Fringe Capacitors," *IEEE Trans. Circuits Syst. I*, vol. 61, no. 8, pp. 2236–2242, Aug. 2014.

[11] S. K. Gonugondla, C. Sakr, H. Dbouk and N. R. Shanbhag, "Fundamental limits on the precision of in-memory architectures", *Proc. ICCAD*, pp. 1-10, 2020.