Robust and High-Performance 12-T Interlocked SRAM for In-Memory Computing

Neelam Surana, Mili Lavania, Abhishek Barma and Joycee Mekie
Department of Electrical Engineering, Indian Institute of Technology, Gandhinagar, 382355, India
(neelam.surana, mili.lavania, barma.abhishek, and joycee)@iitgn.ac.in

***Abstract*—In this paper, we analyze the existing SRAM based In-Memory Computing(IMC) proposals and show through exhaustive simulations that they fail under process variations. 6-T SRAM, 8-T SRAM, and 10-T SRAM based IMC architectures suffer from compute-disturb (stored data flips during IMC), compute-failure (provides false computation results), and half-select failures, respectively. To circumvent these issues, we propose a novel 12-T Dual Port Dual Interlocked-storage Cell (DPDICE) SRAM. DPDICE SRAM based IMC architecture(DPDICE-IMC) can perform essential boolean functions successfully in a single cycle and can perform basic arithmetic operations such as add and multiply. The most striking feature is that DPDICE-IMC architecture can perform IMC on two datasets simultaneously, thus doubling the throughput. Cumulatively, the proposed DPDICE-IMC is 26.7%, 8×, and 28% better than 6-T SRAM, 8-T SRAM, and 10-T SRAM based IMC architectures, respectively.**

***Index Terms*: In-Memory Computing, SRAM, DICE, Dual Port Memory, Interlock Structure**

I. Introduction

State-of-the-art computing systems are based on the Von-Neumann model, which has physically decoupled memory and computing units [1], [2]. However, for data-intensive applications this model can pose severe limitation on the system performance owing to the need for large amount of data transfer between memory and compute units. Subsequently, in-Memory Computing (IMC) has emerged as a promising alternative [2] as it amalgamates compute units within memory, enabling all basic data processing to be done within the memory. This not only improves system performance, but makes it extremely energy-efficient [2]–[6].

Existing work on IMC include *Neural Cache* [2] and [7] where IMC is performed in conventional 6-T SRAM, *XSRAM* [4] where IMC is performed in conventional 8-T SRAM, and *Recryptor* [3], and *Blade* [5] where IMC is performed in 10-T SRAM. However, all these techniques fail under the process variations due to different reasons. Our proposed IMC on DPDICE based SRAM cell mitigates all these issues.

Our Contributions: We have performed extensive analysis and Monte-Carlo (MC) simulations of existing IMC techniques in SRAM and show that they fail due to process variations. We propose a process variation tolerant DPDICE SRAM cell and design an SRAM with IMC. The proposed DPDICE based IMC does not show any compute-error up to 0.6V, computes boolean functions in one cycle, and can be exteded to perform in-memory additions and multiplications. One of the most striking feature of our proposed DPDICE-based IMC is the fact that it allows two in-memory computes in a single cycle due to dual (double) ports, making it even more energy and performance efficient.

The paper organization is as follows. Section II discusses the existing SRAM based IMC architectures. Section III proposes DPDICE SRAM for robust IMC. Section IV discusses the results and Section V concludes the paper.

II. Existing IMC Techniques in SRAM

Several IMC techniques for SRAM have been proposed in the earlier works [2]–[7] and these exhibit functional failures due to process variations as discussed in this section. For this, we have performed statistical simulations to capture the effects of process variations. The Monte-Carlo (MC) simulations reported in this work are done considering 6-σ process variations in 1 KB SRAM at worst case process corner in UMC-65nm technology.

1) IMC in conventional 6-T SRAM: IMC in 6-T SRAM is proposed in [2], [7]. Fig.1 shows the schematic of 6-T SRAM based IMC architecture. Only two SRAM cells have been shown for illustration purpose. To perform compute operation on two operands, first the shared column signals, BL(bitline) and BLB (complemented bitline), are precharged to full-voltage (VDD), and then left floating. The two word lines corresponding to the two operands (here, WL0 and WL1) are asserted simultaneously. Through sense-amplifier(SA) connected to BL, *AND* and *NAND* outputs are obtained, and through SA connected in BLB *NOR* and *OR* outputs are obtained. The *AND* and *NOR* outputs are then logically combined through *NOR* gate to produce an *XOR* output, which is a basic operation in addition.

Our MC simulations show that IMC in 6-T SRAM causes compute-disturb when operands are ‘0’ and ‘1’. During IMC, if Q_0=1 and Q_1=0, both BL and BLB discharges. However, due to process variations, it is possible that one bitline discharges faster than the other. Since, word-lines are ON, this difference can potentially be sensed by the internal storage node(Q and QB) and get amplified by the cross-coupled inverters, causing the data stored at Q to flip. This phenomenon is known as *compute-disturb*. Fig. 2(a) and Fig. 2(b) demonstrate failures due to compute-disturb observed in Q_0 and Q_1, respectively, in 6-T SRAM while performing IMC at the worst-case fast-NMOS slow-PMOS (FNSP) process corner. In [4] aggressive wordline underdrive (for VDD=1.2, WL=0.6V) is done to

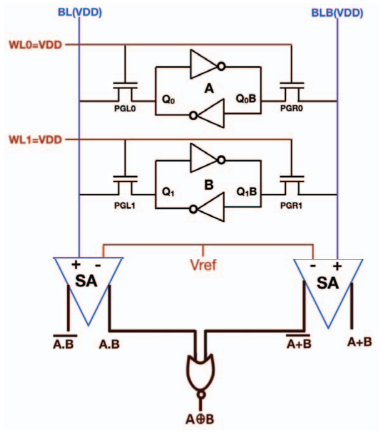

Fig. 1. Schematic of 6-T SRAM based IMC architecture [7]

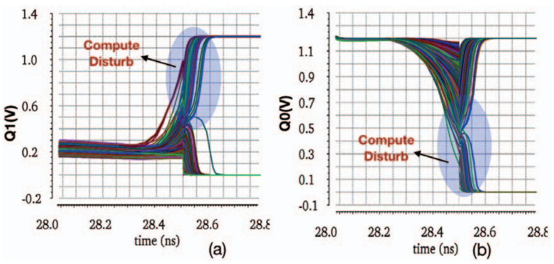

Fig. 2. 1000 MC waveforms of Q node of (a) ROW[1] and (b) ROW[0] for 6-T IMC SRAM at 1.2V

prevent compute-disturb in 6T SRAM, but it costs 3.2× reduction in compute speed.

2) IMC in conventional 8-T SRAM: [4] proposed IMC in conventional 8-T SRAM [1] array. The architecture of the IMC in 8-T SRAM is shown in Fig.3(a). For performing IMC in 8-T SRAM, RWL of two rows are simultaneously turned ON as illustrated in Fig.3(a). IMC in 8-T SRAM is performed by using the pulse width modulation(PWM) of RWL [4]. For a *NOR* computation, RWL pulse width is fixed at 1 ns, and for *NAND*, it is reduced to 0.5 ns. Fig.3(b) shows the timing diagram of *NAND* operation. For the case when A=0 and B=0, i.e. '00', RBL remains at VDD, whereas for the case of '11' RBL discharges to 0 V. However, for '01' and '10' occurrences, RBL which is expected to remain at VDD/2, and which is treated as logic-1 by the skewed inverter used in sense-amplifier, drops due to process variations resulting in compute-failures as shown in Fig. 4. Apart from the compute-failures, the 8-T SRAM based IMC implementation needs carefully designed skewed inverter-based sense amplifier [4] and needs two cycles to obtain one *XOR* operation, as opposed to 1 cycle in [2].

3) Other IMC proposals: [3] and [5] are 10-T SRAM based IMC implementations. The 10-T SRAM cell has read-decoupled structure, and has two read bitlines (RBL and RBLB). Because of the read-decoupled structure of 10-T SRAM and utilization of 2 read bit-lines (RBL and RBLB), the IMC in 10-T SRAM is robust [3]. Also, in 10-T SRAM based IMC architecture [3], *AND*, *NAND*, *NOR*, *OR*, and *XOR* can be computed in a single cycle. However, 10-T SRAM and 8-T SRAM suffer from write half-select failure [1], where the

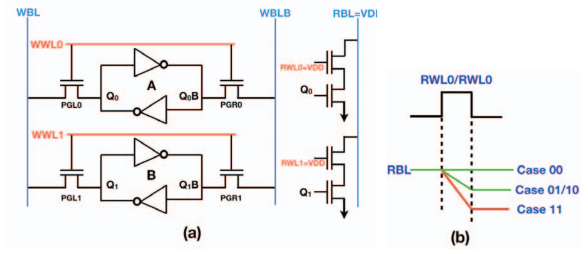

Fig. 3. (a) 8-T SRAM [4] based Architecture (b) Timing diagram of IMC in 8-T SRAM cell for NAND operation (c) Truth table of NAND operation

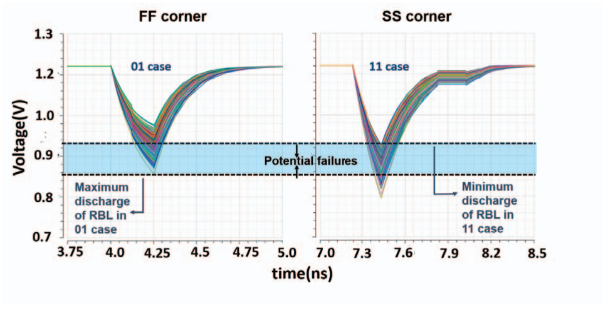

Fig. 4. 1000 MC waveforms of RBL during IMC NAND operation in 8-T SRAM array for 01 case (FF process corner) and 11 case (SS process corner) at 1.2 V

data in unselected cells may be changed when writing on the other selected cells. Some of the other existing IMC in SRAM are based on analog properties such as current comparisons [6], [8]. However, these schemes suffer from significant errors under process variations, especially in lower technologies. Apart from this, the analog IMC method requires complex on-chip analog to digital converter and digital to analog converter circuits.

III. Proposed Dual Port DICE SRAM Cell for IMC

Fig.5 shows the schematic of the proposed DPDICE SRAM cell, which is obtained by appropriately modifying DICE [9] SRAM. In DICE four storage node Q, QB, Q1, and Q1B are used for storing data (Fig.5). To hold the data in DPDICE SRAM, the dual interlocked structure of the conventional DICE SRAM cell [9] is deployed. In the dual interlocked structure, one storage node is controlled by two of its complementary nodes, as shown in Fig.5.

The DPDICE SRAM cell has two word lines (WL_1, WL_2) and four bitlines (BL1, BLB1, BL2, and BLB2), as

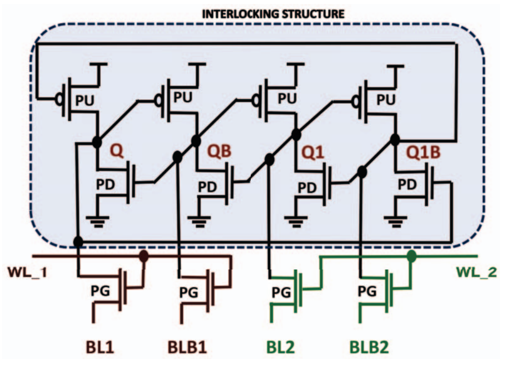

Fig. 5. Schematic of Proposed DPDICE SRAM cell

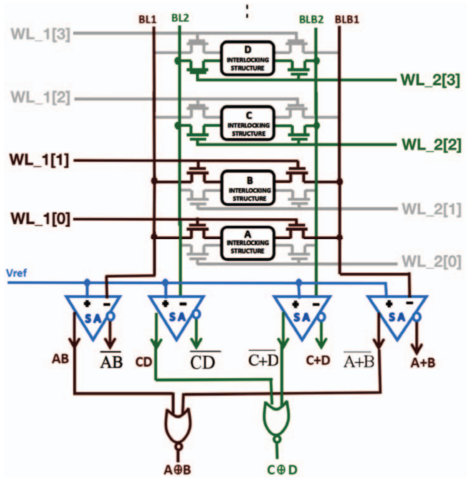

Fig. 6. Block level schematic of IMC in DPDICE SRAM cell

shown in Fig.5. Write in the DPDICE SRAM is similar to that in DICE SRAM [9]. For writing '0', '0' is transferred on BL1 and BL2, and '1' on BLB1 and BLB2. Then both WL_1 and WL_2 are asserted simultaneously to transfer the data from bitlines to storing node(Q, QB, Q1 and Q1B). As DPDICE SRAM has two word lines, in an array, two read operations can be performed simultaneously. To perform the read operation, bit lines are first precharged to VDD. For a single read operation in any row, any of WL_1 or WL_2 of that row can be asserted. The sense amplifier(SA) senses the difference between BL1 and BLB1 and provides the data out. The advantage of activating only one word line (WL_1 or WL_2) is that it prevents the cell from the read-disturb. Even if one node gets upset (changes value), dual interlocked structure of the cell ensures that the other nodes do not change, thus ensures correctness. The interlocking structure also makes the SRAM highly resilient to process variations. Two sets of wordlines allow two independent rows to be read simultaneously in the same cycle.

1) In-Memory Computing In DPDICE: Fig.6 shows the IMC architecture of DPDICE SRAM. All the four bitlines of DPDICE are used for IMC. To carry out Boolean operation between A and B, WL_1[0] and WL_1[1] are activated simultaneously. BL1 computes the *NAND* and *AND* between the Q_A and Q_B, and BLB1 computes the *NOR* and *OR* between the Q_A and Q_B. With the help of an additional *NOR* gate, *XOR* is obtained as shown in Fig.6. In DPDICE SRAM, IMC on the second set of the data can also be achieved using the second set of bitlines, BL2 and BLB2, as shown in Fig.6. BL2 and BLB2 can be utilized for IMC in data vector C and D. So, in the proposed DPDICE SRAM, basic Boolean functions on two different datasets (A, B) and (C, D) can be computed in a single cycle, which is the main feature of the proposed DPDICE SRAM based IMC.

2) Error Resiliency of DPDICE SRAM during IMC: DPDICE SRAM is highly resilient to process variations due to its non-conventional interlocking structure [9]. Fig.7(a) shows the effect of the voltage glitch on Q node. Assume that the data stored in the cell was Q=1. With the help of external current source, Q node is temporally driven to '0'. As can be seen in Fig.7(a), the voltage glitch at the Q node could not affect the QB node, as it is driven by Q1 node, which remains unaffected. As QB and Q1B nodes are unaffected, Q node is restored to '1' as shown in Fig.7(a). Thus, voltage glitch in the single node of DPDICE SRAM can be handled by interlocked structure of the cell.

Fig.7(b) and Fig.7(c) show the robustness of DPDICE-SRAM during compute under process variations. These are the plots obtained at the Q node when 1000 Monte-Carlo IMC simulations are performed in row[1] and row[0]. It can be seen from the Fig.7(b) and Fig.7(c), no data flip is observed during IMC in DPDICE SRAM cell. We further test for the half-select stability of the DPDICE-SRAM under process variations. Fig.7(d) shows the plot of Q node when 1000 Monte-Carlo IMC simulations are carried out between two rows and only half rows are selected. As shown in Fig.7, no half select failure was found in the DPDICE SRAM cell. To summarize, DPDICE SRAM based IMC architecture is able to mitigate all the issues observed in the previous IMC proposals and which doubles the throughput.

3) In Memory Arithmetic Operation in DPDICE: As discussed, IMC in DPDICE can provide basic boolean functions in a single cycle. Once the *XOR* between two inputs are obtained at the output, a single bit *full adder* can be realized with the help of additional circuitry in a way similar to that reported in [2], where data on which computation is to be done is stored in the transposed form. Using this approach, the performance can be significantly improved as an n-bit adder can be implemented in n-cycles [2]. For a k-column SRAM array, k number of additions can be performed in parallel in 'n' cycles. Further, with the help of addition operation, multiplication can be performed in array. Multiplication in the array takes n^2+3n-2 cycles, but at the same time k number of multiplication can be performed in parallel. With the help of *XOR* observed at the output, n-bit division and subtraction can be obtained in $1.5n^2+5n$ and 2n cycles, respectively, with the algorithm discussed in the [2]. In DPDICE, for k-columns in the array, 2k number of arithmetic operations can be performed in parallel as it can provide the boolean operation on the 2 data-sets simultaneously.

IV. Results and Discussion

Reliability remains the main concern in existing IMC methods in SRAM. To find the error during IMC, we measure the compute error rate (CER) of the existing IMC SRAM techniques at different voltage. We simulated the SRAMs for 10000 MC simulations considering standard 6-σ process variations. Fig.8(a) shows the CER comparisons of 6-T SRAM [7], 8-T SRAM [4], 10-T SRAM [3], [5] based IMC architectures and the proposed DPDICE IMC architecture. As can be seen in Fig.8(a), IMC in 10-T SRAM and DPDICE SRAM is error-free and shows 0 CER even at 0.6 V. However, 10-T SRAM suffers from the half select failures.

$$FOM = \frac{(1-CER)\times throughput}{Area \times Energy} \quad (1)$$

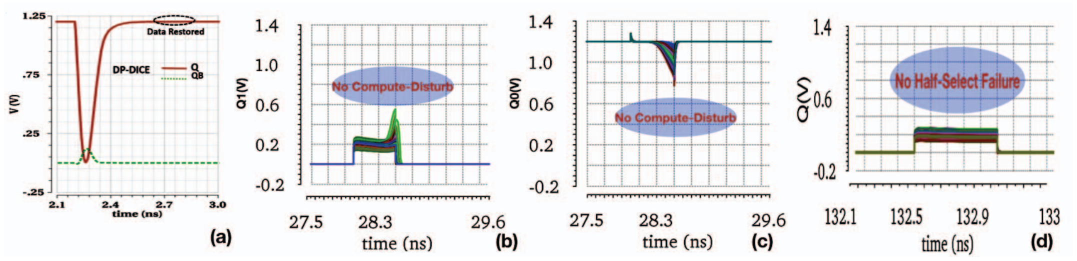

Fig. 7. (a) Effect of a voltage glitch on the stored data in the DPDICE SRAM cell (b) 1000 MC waveforms of a cell in row[0] in DPDICE during IMC (c) 1000 MC waveforms of a cell in row[0] in DPDICE during IMC (d) 1000 MC waveforms of a half-selected cell in DPDICE SRAM

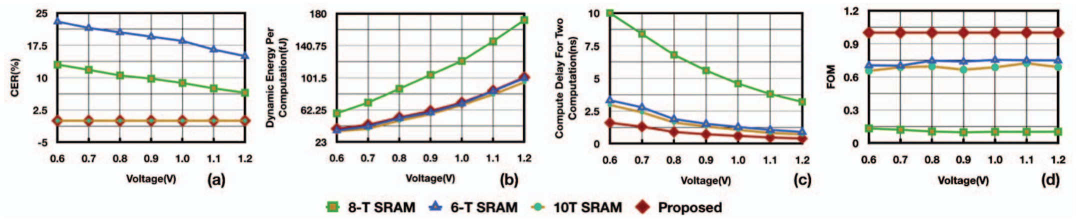

Fig. 8. Comparison results of proposed IMC architectures with existing SRAM based IMC architectures (a) Computation error rate versus voltage (b) Dynamic energy per computation versus voltage (c) Computation Delay for two computation versus voltage (d) FOM comparison normalized to the proposed case

TABLE I
PERFORMANCES COMPARISON SUMMARY

	[2]	[4]	[3]	Proposed
Area per Cell(μm^2)	1.03	1.38	1.74	**2.06**
Compute Disturb	**Yes**	No	No	No
Compute Failure	No	**Yes**	No	No
Half Select Failure	No	**Yes**	**Yes**	No
Data Read Per Cycle	**1**	**1**	**1**	2
Computes Per two Cycles	**2**	**1**	**2**	4

We have compared the proposed IMC architectures with existing IMCs in terms of dynamic energy required per computation and computing delay per two computations. Two computations are considered as 8T IMC SRAM needs 2 cycles for a single compute. As shown in Fig.8(c), DPDICE SRAM has almost similar dynamic energy requirement compared to 10-T [3] and 6-T [7]. Further, when compared to the computing in 8-T SRAM [4], the proposed technique consumes at least ∼57% lesser dynamic energy per computation. In terms of the speed, DPDICE has ∼48% lesser compute delay per two computation than 10-T SRAM [3], [5]. Table I summarizes the comparison in IMC architectures. As seen in Table I, DPDICE SRAM takes almost ∼16% more area than 10-T SRAM. Layouts are drawn using the vertical n-well technique. To have a cumulative comparison, we have used a *figure of Merit(FOM)* as defined in (1). As it can be seen from Fig.8(d), proposed technique performs cumulatively better than existing SRAM based IMC techniques [2]–[4].

V. Conclusion

We have performed an in-depth study of existing SRAM based In-Memory Computing(IMC) architectures. We show through exhaustive simulations that existing SRAM based IMC architectures fail under process variations. To have a robust IMC in SRAM, we proposed a novel 12-T SRAM cell, Dual Port Dual Interlocked Storage Cell (DPDICE). Overall, the DPDICE SRAM based IMC architecture does not show any failure even at 0.6 V with process variations, can potentially give double throughput and has lower energy requirements. We show how two in-memory computes can be simultaneously performed using DPDICE SRAM, and in future we would like to extend the DPDICE IMC to support floating point arithmetic.

Acknowledgment

This work is supported by a grants received from the Ministry of Electronics and Information Technology (MEITY), Govt. of India for SMDP- C2SD project and under Visvesvaraya Ph.D. scheme, and by grant received from the Department of Science and Technology (SERB) number CRG/2018/005013.

References

[1] N. H. Weste and D. Harris, *CMOS VLSI design: a circuits and systems perspective*. Pearson Education India, 2015.

[2] Eckert et al., "Neural cache: Bit-serial in-cache acceleration of deep neural networks," in *Proceedings of the 45th Annual International Symposium on Computer Architecture*, 2018, pp. 383–396.

[3] Y. Zhang, L. Xu, Q. Dong, J. Wang, D. Blaauw, and D. Sylvester, "Recryptor: a reconfigurable cryptographic cortex-m0 processor with in-memory and near-memory computing for iot security," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 4, pp. 995–1005, 2018.

[4] A. Agrawal, A. Jaiswal, C. Lee, and K. Roy, "X-sram: Enabling in-memory boolean computations in CMOS static random access memories," *IEEE Transactions on Circuits and Systems I: Regular Papers*, no. 99, pp. 1–14, 2018.

[5] W. A. Simon, Y. M. Qureshi, A. Levisse, M. Zapater, and D. Atienza, "Blade: A bitline accelerator for devices on the edge," in *Proceedings of the 2019 on Great Lakes Symposium on VLSI*. ACM, 2019, pp. 207–212.

[6] A. A. et al., "Xcel-ram: Accelerating binary neural networks in high-throughput sram compute arrays," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 8, pp. 3064–3076, Aug 2019.

[7] S. Jeloka, N. B. Akesh, D. Sylvester, and D. Blaauw, "A 28 nm configurable memory (tcam/bcam/sram) using push-rule 6t bit cell enabling logic-in-memory," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 4, pp. 1009–1021, 2016.

[8] A. Jaiswal, I. Chakraborty, A. Agrawal, and K. Roy, "8T SRAM cell as a multibit dot-product engine for beyond von neumann computing," *IEEE Transactions on VLSI Systems*, pp. 1–12, 2019.

[9] T. Calin, M. Nicolaidis, and R. Velazco, "Upset hardened memory design for submicron cmos technology," *IEEE Transactions on nuclear science*, vol. 43, no. 6, pp. 2874–2878, 1996.