Resonant Compute-In-Memory (rCIM) 10T SRAM Macro for Boolean Logic

Dhandeep Challagundla[1], Ignatius Bezzam[2], Biprangshu Saha[3], Riadul Islam[1]
[1]*University of Maryland, Baltimore County,* [2]*Rezonent Inc.,* [3]*Plus PPA Nanochip Technologies*
vd58139@umbc.edu, i@rezonent.us, itsbiprangshu@gmail.com, riaduli@umbc.edu

2023 IEEE 41st International Conference on Computer Design (ICCD) | 979-8-3503-4291-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICCD58817.2023.00026

***Abstract*—Traditional State-of-the-Art computing platforms have relied on silicon-based static random access memories (SRAM) and digital Boolean logic for intensive computations. Although the metal-oxide-semiconductor transistors have been aggressively scaled, the fundamental von-Neumann computing architecture has remained unaltered. The emerging paradigm of Compute-in-Memory (CIM) offers a promising solution to overcome the memory wall bottleneck in traditional von-Neumann architectures by enabling the processing and storing of information within SRAM memory elements. This article introduces an energy-recycling resonant 10T-SRAM architecture that facilitates in-memory computations to minimize the need for data movement between the processing core and memory. Series resonant write driver is utilized to efficiently recycle the discharged energy during a writing operation to reduce the overall energy consumption of the SRAM architecture. The feasibility of the proposed rCIM has been demonstrated by implementing it on an 8KB memory array using TSMC 28nm PDK. Additionally, a comprehensive Monte Carlo variation analysis was conducted to ensure the robustness and reliability of the scheme under process variations. To demonstrate the effectiveness of the proposed architecture, we evaluate its performance using the EPFL combinational benchmark suite. The proposed Resonant Compute-In-Memory (rCIM) consumes 55.42% lower energy than standard von-Neumann architecture and achieves a throughput of 88.2-106.6 GOPS/s.**

***Index Terms*—Static Random Access Memory (SRAM), compute-in-memory (CIM), Von-Neumann memory bottleneck, series LC resonance.**

I. Introduction

While the von-Neumann architecture has played a significant role in the rapid advancement of computing over the past seven decades, it is not without limitations. One major drawback is the architecture's reliance on data movement between processors and memory elements, which results in increased latency and energy consumption [1].

The traditional von-Neumann architecture separates the arithmetic logic unit (ALU) from the memory, as depicted in Fig. 1 (a). The need for frequent data transfers between the two components during operations leads to high energy consumption and latency, known as the memory-wall bottleneck [2]. As shown in Fig. 1 (b), the memory fetch operation consumes >99% of the total energy in a traditional von-Neumann architecture, compared to a 32-bit ALU addition operation consuming <1% of the total energy [3].

This work was supported in part by Rezonent Inc. under Grant CORP-0061, National Science Foundation (NSF) award number: 2138253, and UMBC Startup grant.

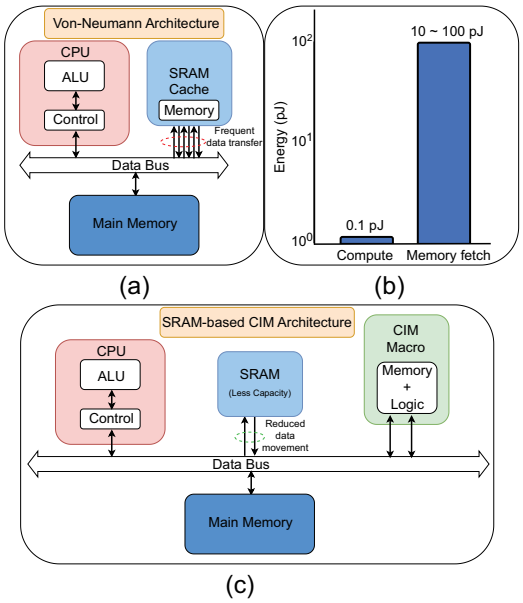

Fig. 1: (a) Traditional von-Neumann architecture creates a memory wall bottleneck caused by frequent data transfer between CPU and cache memory, (b) the memory fetch operation consumes 99% of the energy in a system compared to a 32-bit ALU addition [3]. (c) Compute-in-memory (CIM) architecture mitigates the need to frequent data transfer by incorporating logic into memory.

Compute-In-Memory (CIM) architecture, shown in Fig. 1 (c), emerges as a highly promising solution for minimizing data movement, enhancing computational throughput, and improving energy efficiency in data intensive applications [4]. Till date, the CIM concept has been extensively discussed and explored across various memory technologies, including static random access memory (SRAM) [2], dynamic random access memory (DRAM) [5], and emerging non-volatile memory, spin-transfer-torque (STT) MRAM [6], Resistive random access memory (RRAM) [7], flash memory [8], etc. This paper uses SRAM-based CIM architectures, as SRAM cells have high reliability, low latency, and are robust to process variations.

SRAM-based CIM architectures have been heavily investigated to perform various operations, such as Multiply-and-

2576-6996/23/$31.00 ©2023 IEEE
DOI 10.1109/ICCD58817.2023.00026

Accumulate (MAC) operations [9]–[14], boolean logic operations [15]–[20] and content-addressable memory (CAM) operations [3], [4], [21], [22] for fast searching applications.

In this work, we implement an SRAM-based energy-recycling Resonant CIM (rCIM) to enable boolean logic operations within memory to increase the throughput and energy efficiency of the system. In addition to performing boolean computations within memory, this work also uses an energy-recycling low-power technique, namely, a series LC resonance scheme that stores the discharged energy using an on-chip inductor and recycles the energy back into the design. In particular, the main contributions of this work are:

- First-ever CIM architecture utilizing series LC resonance scheme to recycle discharged energy.
- A novel dual-read port 10T SRAM cell that enables multi-row and multi-column computations during a single access cycle.
- Demonstration of the proposed architecture for various combinational circuits using EPFL benchmark suite.

The remainder of the document is organized as follows. Section II discusses state-of-the-art CIM techniques and fundamental series resonance operations. Section III proposes the energy-recycling resonant compute-in-memory (rCIM) architecture and details the working of SRAM mode, logical operations, and energy-recycling write driver design. The validation of the proposed rCIM architecture is discussed in Section IV using EPFL combinational benchmark suite and Section V concludes the work.

II. Background

This section presents the existing high-performance CIM techniques and design challenges. Many compute-in-memory (CIM) techniques have been presented lately.

A. Existing CIM architectures

In recent years, significant work has been reported to alleviate the memory bottleneck of traditional von-Neumann architectures using SRAM-based CIM architectures [2]. In [4], [10], [11], [13], [15], [18], [21] traditional 6T SRAM cells are repurposed to perform logical operations, binary/ternary content addressable memory (CAM) operations and, multiply-and-accumulate (MAC) operations. Such operations require multi-row activations to access multiple operands in a single access cycle. However, the conventional CIM architectures based on 6T bitcells encounter significant reliability concerns, particularly read disturbance, when multiple rows are activated simultaneously [3], [18]. [21] uses a wordline-underdrive scheme to mitigate the bit-flipping caused by the activation of multiple rows but at the cost of performance loss, whereas, [4] uses a dual-split-VDD scheme but requires a high engineering effort.

In [23] a read-decoupled 8T SRAM-based CIM was introduced to mitigate the read disturbance caused due to multirow activations. This technique necessitates storing data in columns, which leads to increased implementation complexity in SRAM write operations and requires data movement. In [24], an 8T SRAM structure that allows for full array computing and in-situ storage is proposed, however, this decreases the storage density of the SRAM array and the operands are unable to perform independent operations [25]. In [20], a transposable 8T bitcell is implemented for multioperand bitwise boolean logic operations. However, it still has the read-disturb issue like the traditional 6T bitcell [3].

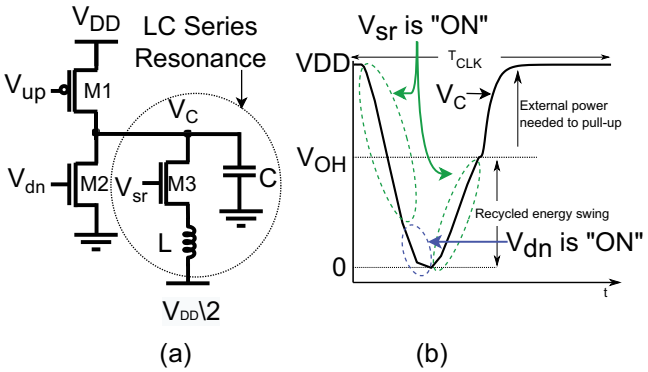

Fig. 2: (a) Series LC resonance topology when the inductor "L" is connected in series to a load capacitance "C" (b) Output at capacitor node V_C shows the recycled energy swing from "0" to V_{OH}

In order to reduce the need for data movement in single column computations, [17] proposes a two-direction 10T SRAM cell with decoupled horizontal and vertical read ports. This architecture uses one sense amplifier for every row and column. This causes a huge area overhead. A CIM macro with cross-coupled 10T cell structure is proposed in [25] for computational results to be stored in situ. This implementation performs twin-operand computation but requires high engineering effort due to its analog nature. In order to overcome unnecessary data movement due to a single column or a single row constraint, this work proposes a dual read port 10T SRAM cell that has separate read bitlines for each vector operand. This allows the data stored in different columns and different rows to be accessed during a single access cycle without the requirement of moving data which is not supported by existing CIM architectures. Moreover, the proposed 10T SRAM bitcell enables series LC resonance operations to recycle the discharged energy. This greatly reduces the dynamic power consumption during write operations.

B. Series LC resonance

Researchers have identified that a significant amount of dynamic and static power is consumed by SRAM write operations [26]. Among several low power clocking schemes [27]–[29], series LC resonance demonstrates prominent savings in dynamic power consumption [30], [31]. This paper incorporates series LC resonance in the write operation of SRAM. Whenever a signal goes from logic "1" to "0," the energy is dissipated in the form of heat. LC resonance techniques harvest this dissipated energy and recycle it back into the design. There are two kinds of LC resonance techniques, namely, parallel resonance [28] and series resonance [32]. In parallel

resonance [29], an on-chip inductor is placed in parallel to the load capacitance. This technique saves the highest amount of power only when the system clock frequency matches the resonant frequency. To overcome the narrow frequency band of operations, researchers utilize series resonance techniques [32].

Fig. 2 (a) shows a series LC resonant circuit with an inductor size "L" and a load capacitance "C" [32]. During resonance operation, first the V_{sr} signal goes high, during which the inductor stores the dissipated energy from the load capacitance "C." Then the V_{dn} signal is asserted to completely discharge the V_C node. During the energy-recovery phase, the V_{sr} signal goes high again, this is when the energy stored in the inductor is recycled back into the V_C node. Fig. 2 (b) shows the output at V_c node. The free energy swing obtained as a result of recycling dissipated energy is the difference between "0" and V_{OH}. Then the V_{up} signal is asserted to pull up the V_C node to VDD.

This work utilizes a write driver based on series resonance to effectively harvest some of the discharged energy during an SRAM write operation and recycles the energy back during the pre-charge phase, significant reducing the dynamic power consumption of the system.

III. Proposed CIM Architecture

A conventional CIM architecture consists of a traditional SRAM cache that supports other computations inside the same macro. This Section will discuss the conventional SRAM read & write operations and our proposed CIM architecture with other logical operations.

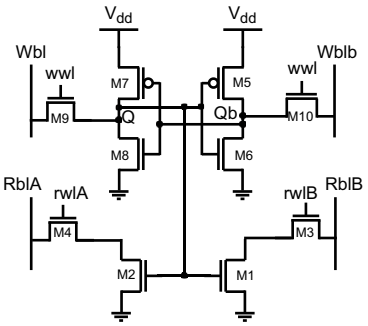

Fig. 3: The 10T SRAM cell consists of 4 additional transistors to enable robust in-memory computation.

Conventional Read & write: The proposed 10T SRAM bitcell shown in Fig. 3 comprises the standard 6T bitcell enhanced with an additional decoupled dual read-port formed by transistors $M1 - M4$. The write operation is performed similarly to a 6T bitcell, by activating the access transistors $M9 - M10$ using write-wordline (wwl), whereas the read operation is performed by activating a read-wordline ($rwlA$) while keeping the wwl low. During the read operation, the $RblA$ is initially precharged to VDD, and if Q = "1", the $RblA$ is discharged. On the other hand, if Q = "0", $RblA$ stays at its initial precharged voltage. To sense the voltage, the $RblA$ is connected to a sense amplifier.

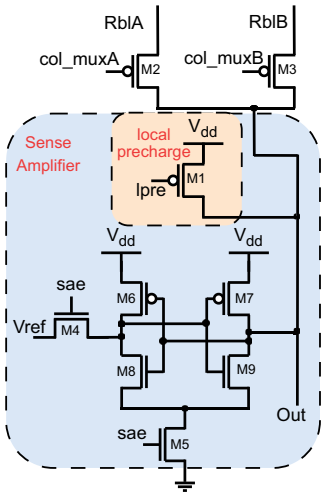

Fig. 4: The single-ended sense amplifier has a local precharge transistor $M1$ to precharge the sensing end to VDD along with an $M4$ switch to provide a $Vref$ voltage for comparison of the discharged voltage of the $RblA/RblB$.

Fig. 4 shows the circuit of a single-ended sense amplifier adapted from [33]. The single-ended sense amplifier is connected to the Rbl on one side and a reference voltage, positioned at $VDD/2$, on the other end. Depending on the voltage value of the Rbl, the sense amplifier toggles the "Out" node to a "1" or a "0." Initially, transistor $M1$ is enabled to precharge the "Out" node to "VDD" using the local precharge signal "$lpre$." Now, the transistor $M1$ is turned "OFF" and the column mux transistors $M2$ and $M3$ are enabled to discharge the $RblA/RblB$ according to the operation performed. Only a single $RblA/RblB$ is connected to the sensing end during a conventional read. Once the charge sharing is complete, we turn "OFF" $M2$ and $M3$ transistors, and the sense amplifier enable ("sae") is activated. The transistor $M4$ connects the appropriate $Vref$ voltage to one end of the sense amplifier, and if the voltage of "Out" is less than $VDD/2$, then the discharge path $M9$, $M5$ dominates and completely discharges "Out" node, to result in a "0" output. For a read operation, the output of the sense amplifier is inverted and provided to the output read port. The decoupled $RblA$ and $RblB$ in the design of the 10T bitcell enables a large voltage swing during the read operation, eliminating any potential read disturb failures. The transistors $M1 - M4$ are also implemented at minimum sizes to minimize the area overhead associated with the 10T bitcell.

NAND Operation: A single NAND and NOR operation is depicted in Fig. 5. The output of a NAND operation is "0" only if both the inputs are "1". Unlike [23], by incorporating two read ports, the system enables the independent access of two vector operands, without limiting the computation to a single row or column of the SRAM array. This feature allows for more flexible and efficient data retrieval, enhancing the overall functionality and performance of the system. Consider activating $rwlA$ and $rwlB$ simultaneously corresponding to the rows storing vector operands "A" and "B." The initially precharged $RblA$ and $RblB$ will eventually discharge to 0 V

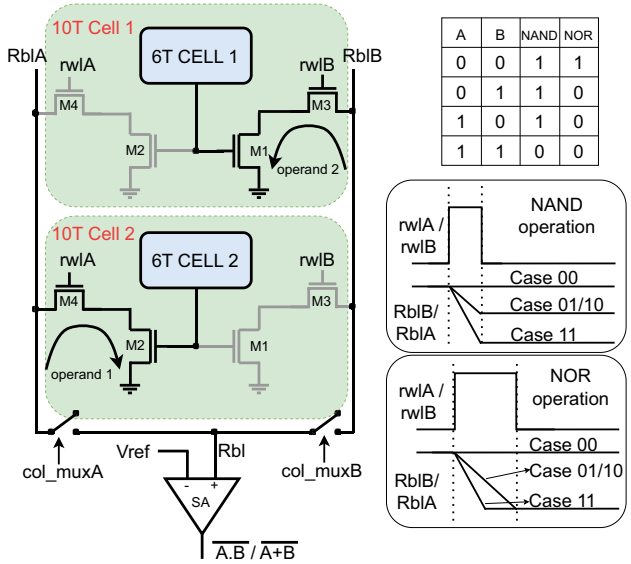

Fig. 5: Circuit schematic of 10T-SRAM for implementing a single NAND/NOR operation with equivalent circuit traced by transistors $M1 - M4$ while data is read from Cell 1 and Cell 2.

if any of its corresponding operands store a "1." However, the fall time of $RblA/RblB$, curtailed during computation, and strongly depend on whether one bitcell is storing a "1" or if both the bitcells are "1." The $rwlA$ & $rwlB$ are timed to leverage this discharge rate ensuring that the $RblA/RblB$ does not discharge completely in cases "10/01". This allows differentiating the voltage levels on $RblA/RblB$ for two cases "10/01" and "11." The $RblA/RblB$ will not discharge if the Q is "0" for both the operands "A" and "B." We have performed extensive Monte-Carlo simulations to analyze and identify the discharge rate of $RblA/RblB$. The results of these simulations are discussed in Chapter. IV C. When using a single-ended sense amplifier with a $Vref$ value lower than the typical "10/01" case $RblA/RblB$ discharge, the output of the sense amplifier will result in the output mimicking the NAND operation by producing a logical "1" for all three cases ("00" and "10/01").

A single-ended sense amplifier with a $Vref$ value lower than the typical "10/01" case, the $RblA/RblB$ discharge will output the NAND result as "1" for the three cases ("00" and "10/01"). Thus the output mimics the NAND operation. The same NAND implementation can also perform a NOT operation by providing a single vector operand to the $rwlA/rwlB$. Thus, performing a NAND operation that only considers two cases ("00" and "11") would result in an inversion operation.

NOR Operation: A NOR operation outputs a "1" only if the two operands "A" and "B" are "0." When we activate two $rwlA/rwlB$, the $RblA/RblB$ discharge rate depends upon the three input cases ("10/01" and "11"). Activating the $rwlA/rwlB$ for a larger time allows the $RblA/RblB$ to completely discharge to 0V for the "10/01" cases, resulting in a "0" output. The same sense amplifier can be reused to mimic the NOR operation. The timing of the $rwlA/rwlB$ is

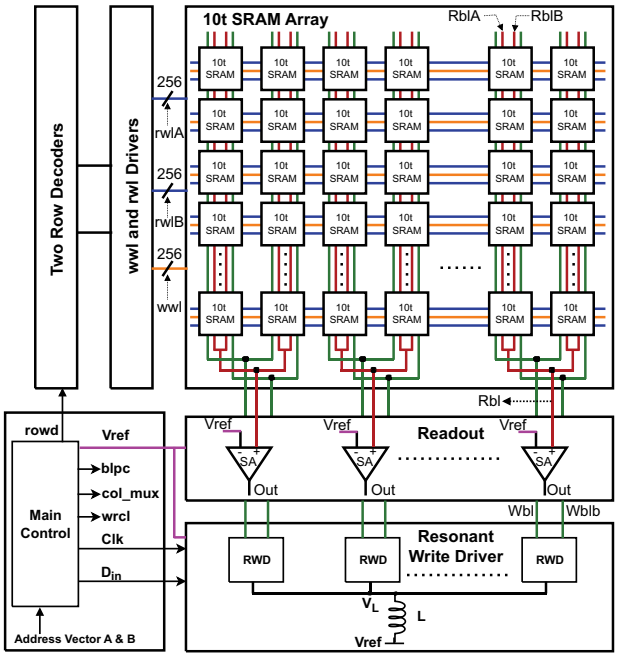

Fig. 6: The proposed rCIM comprises 10T SRAM arrays, a resonant write driver, a readout circuitry for boolean computation, and a write-back circuit to store the computed data within the SRAM array.

implemented using a simple delay circuit and characterized precisely to generate two different pulse widths for NAND and NOR cases.

Proposed rCIM: The overall structure of the proposed rCIM 10T SRAM macro, shown in Fig. 6, consists of a 10T SRAM array, a resonant write driver, a readout circuit for boolean computation, row decoders for write wordline (wwl) and read wordlines ($rwlA/rwlB$) and peripheral control circuit to generate internal signals.

The 10T SRAM array consists of 256×256 arrays to store and read data using vertical bitlines and horizontal wordlines. During a conventional read/write operation, only one of the two-row decoders is enabled to generate the write wordline (wwl). However, when performing boolean operations, both row decoders are enabled simultaneously. This allows for simultaneously activating wordlines corresponding to the two input operands "A" and "B." The pulse width of the clock controls the pulse width of the wwl and the pulse width of the $rwlA/rwlB$. A buffer-based delay circuit is incorporated with the clock to generate the required rwl pulse width. Different pulse widths can be achieved by varying the number of buffers in the delay circuit, which are necessary for performing NAND and NOR operations.

The SRAM array has a column MUX factor of 2, indicating that every two columns are interconnected and share a single sense amplifier. This design allows for the efficient utilization of sense amplifiers in the readout circuit. In this configuration, a single-ended sense amplifier is assigned to every two columns of the bitcell array, enabling both conventional read

operations and effective computations.

The output of the sense amplifier is latched and utilized as input data for the subsequent clock cycle. This input data is fed into an energy-recycling resonant write driver (RWD), which facilitates the writeback of the computation output into a bitcell. The architecture is capable of executing 128 operations in parallel. Considering that the output size is small compared to the full SRAM size (256×256), we have organized the data by columns based on hierarchical demands. Ensuring a maximum of 128 gates in each stage of the combinational circuit will optimize the architecture's efficiency. The energy-recycling RWD, shown in Fig. 7, uses an on-chip inductor, in series resonance topology, to store the discharged energy from write bitlines (Wbl and $Wblb$) and recycles it back into the write bitlines. By doing so, the system minimizes power consumption during the precharge phase. The on-chip inductor is conditionally connected to the bitlines on one end and to a bias voltage ($VDD/2$) on the other end. When the bitlines transition from a logic "1" to "0", the discharged energy is stored in the $VDD/2$ node. During the energy-recovery phase, the series resonance circuit empties the charge from $VDD/2$, leaving zero net currents from that node for the whole cycle.

The write bitlines (Wbl and $Wblb$) are connected to the series inductor through transistors $M9 - M12$ controlled by vsr_d and vsr_{db} signals generated from the input clock signal. Depending on the input data provided, we either discharge Wbl if the input data is "0" or $Wblb$ if otherwise. The transistors $M7$ and $M8$ are used to discharge the write bitline, ensuring a full rail-to-rail swing. Once the write operation is done, we enable the same transmission gate as before to restore the stored energy from the inductor. This reduces the overall power consumption required to precharge the bitlines for the next operation, as the bit lines are not needed to be precharged from "0."

For a given resonant frequency, the product of inductance and load capacitance stays constant. Thus, sharing a single inductor for all the write drivers greatly reduces the size of the inductor as the load capacitance increases. We require N number of write drivers for N number of bits. Hence, the load capacitance increases N times while the effective resistance path decreases N times.

IV. Results and Discussion

A. Experimental Setup

We used TSMC 28N technology to implement the proposed rCIM architecture. The 10T-based SRAM cache memory size is 8KB and uses Cadence Virtuoso and Specter simulator for implementation and analysis. Validation of the proposed rCIM architecture is demonstrated using EPFL combinational benchmark suite [34] synthesized with YOSYS logic synthesizer [35]. ASIC implementations obtained post place and route from Synopsys ICC2 are used to compare with the proposed rCIM implementations. The 10T SRAM macro consists of 256×256 bitcells, with a column MUX factor of 2. It also includes a Readout circuit comprising 128 Sense Amplifiers and an energy-recycling write driver comprising 128 resonant write drivers sharing a single series inductor. Two-row decoders are used to decode the input address from the CPU.

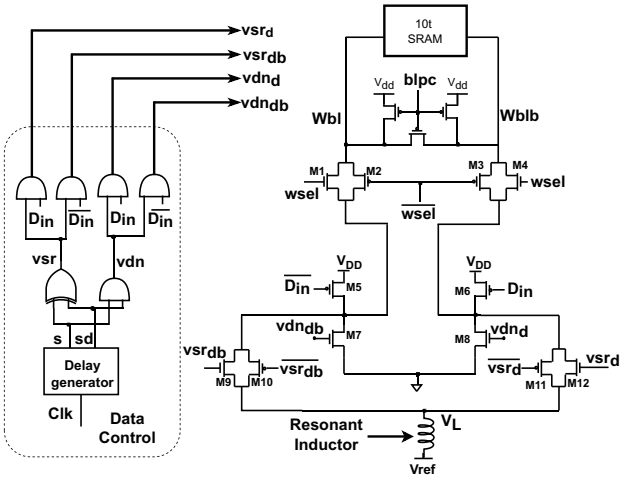

Fig. 7: The energy-recycling resonant driver consists of a data control unit to enable the write operation and an on-chip inductor that stores the discharged energy during a write operation and recycles it during the precharge phase.

Depending on the compute or the read/write operations, the wwl/rwl drivers are enabled.

B. Functionality

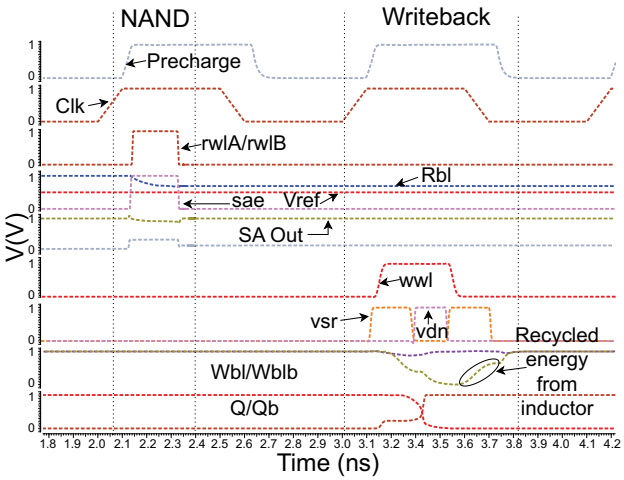

Fig. 8: The functionality waveform depicts a NAND computation of "01"/"10" data and writeback operation showing the successful writing back of the result into the memory and recycling energy during the writeback operation.

Fig. 8 shows the transient simulation for a NAND operation of 2 inputs, "1" & "0", with a reference voltage ($Vref$) value of $VDD/2$. The rCIM operates in 2 clock cycles. The first clock cycle performs the boolean operation (NAND/NOR), shown in Fig. 5 and the second clock cycle completes the writing of the output from the previous clock cycle using the resonant write driver shown in Fig. 7. When the CPU sends the compute instruction, the two input vector operand addresses are provided to row decoders, that enable the

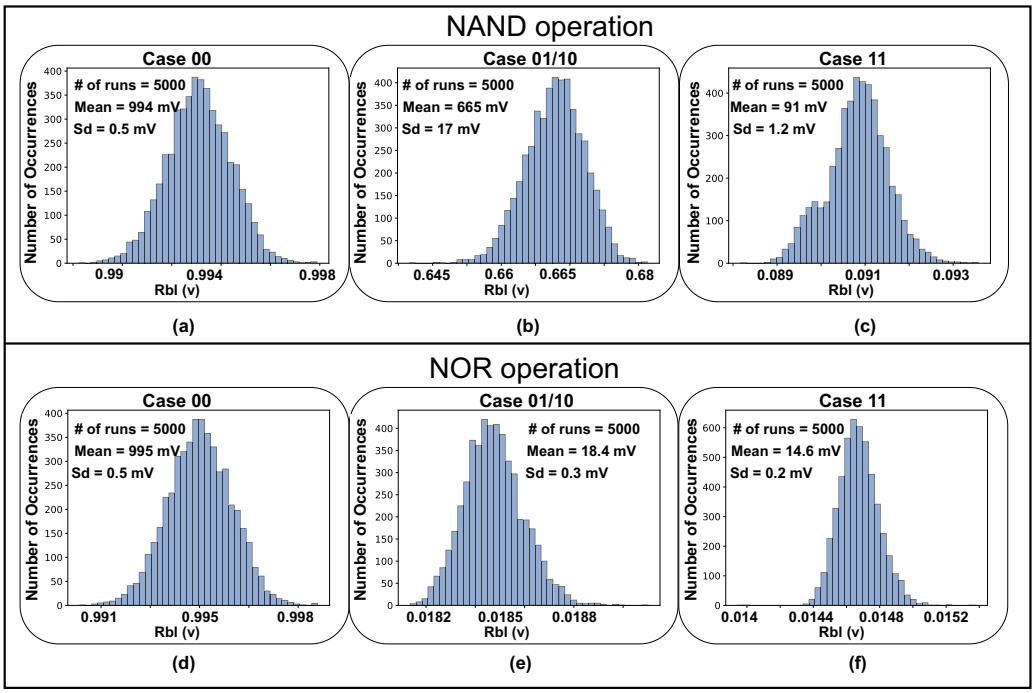

Fig. 9: Illustration of the robustness of the proposed rCIM using Monte-Carlo simulation results for different input vector cases considering 5000 samples with $\pm 10\%$ length variation.

$rwlA/rwlB$ signals. The 10T SRAM cell storing "1" will create a discharge path for the Rbl, which are connected to sense amplifiers ($SA's$) using a column mux for vector operand "A" and "B". The Rbl signal will discharge to 665 mV for inputs "1" & "0". This value is quite greater than the $Vref$ voltage, which is 500 mV ($VDD/2$). Thus, the output signal Out of the sense amplifier outputs a "1" representing the NAND operation of "0" and "1".

The output of the sense amplifier is latched onto the data latch and is written into the destination address on the next clock cycle. On the positive edge of the clock, the wwl is asserted using a single row decoder. We generate two voltage pulses vsr and vdn signals from a signal s, and its delayed version sd. The vsr signal enables series resonance, by conditionally connecting the $Rbl's$ to the series inductor through transmission gates $M9 - M12$ (Fig. 7). After successfully writing the data into the bitcell, the $Wblb$ is precharged using the energy stored in the inductor.

For measuring the performance and functionality of the proposed 10T bitcell under process variations, we considered 5000 samples of Rbl discharge using Monte-Carlo simulations performed using Hspice. $\pm 10\%$ variation in the length of all devices is considered while performing the simulations. The Rbl discharge distribution values for different vector input cases are shown in Fig. 9. For the NAND operation, the Rbl has a mean discharge value of 0.665 V with a standard deviation of 17 mV for case "01/10," shown in Fig. 9 (b). For NAND operation with case "00," shown in Fig. 9 (a), the mean discharge value of Rbl is 994 mV with a standard deviation of 0.5 mV. For NAND operation with "11" case (Fig. 9 (c)), the Rbl has a mean discharge value of 91 mV with a standard deviation of 1.2 mV. For the NOR operation with "00" case (Fig. 9 (d)), the Rbl has a mean discharge value of 995 mV with a standard deviation of 0.5 mV. For the NOR operation, the Rbl has a mean discharge value of 18.4 mV with a standard deviation of 0.3 mV for case "01/10," shown in Fig. 9 (e). For the NOR operation with "11" case (Fig. 9 (f)), the Rbl has a mean discharge value of 14.6 mV with a standard deviation of 0.2 mV. These statistical values enable the characterization of the reference voltage ($Vref$) used in sense amplifier (Fig. 4).

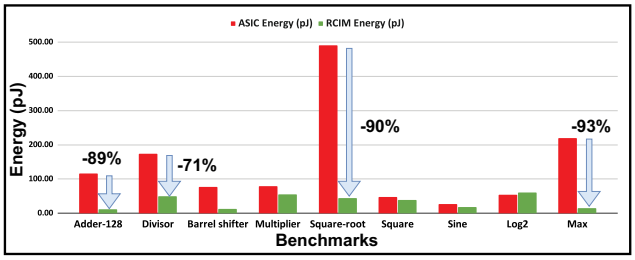

Fig. 10: Illustration of rCIM and ASIC implementations in terms of energy consumption depict 90% lower energy consumption for square-root benchmark circuit while reducing 71% of energy consumption for divisor circuit.

Table. I depicts the benchmark results implemented on the proposed rCIM architecture compared with the ASIC implementations. The ASIC synthesis results were obtained post place and route (PNR) from Synopsys ICC2. EPFL combinational benchmark suite was considered for benchmarking the

TABLE I: When comparing the proposed rCIM macros energy consumption with the ASIC implementations for EPFL combinational benchmark suite, the proposed architecture demonstrated a 55% reduction in average energy consumption.

Benchmark	# of Input Bits	# of Output Bits	# of NAND Gates	# of NOR Gates	# of Inverters	rCIM Latency (ns)	rCIM Energy (pJ)	ASIC Latency (ns)	ASIC Energy (pJ)	rCIM Energy Savings
Adder-128	256	129	170	1102	271	6.05	12.13	6.76	115.79	89.52%
Divisor	128	128	6699	18416	7732	88.60	50.13	77.92	174.18	71.22%
Barrel shifter	135	128	1866	1086	7	8.25	13.01	2.47	77.02	83.11%
Multiplier	128	128	6538	20525	8607	97.45	54.80	7.51	79.25	30.86%
Square-root	128	64	5356	17754	4174	78.55	44.65	89.19	490.46	90.90%
Square	64	128	3077	14002	6285	65.50	38.30	7.99	48.04	20.27%
Sine	24	24	2321	4065	1153	21.20	18.28	5.16	27.29	33.01%
Log2	32	32	10363	22299	7714	108.15	60.80	9.61	53.64	-13.33%
Max	512	130	667	2353	1157	12.75	14.81	9.07	219.29	93.25%
Average	**≈157**	**99**	**≈4118**	**≈11290**	**≈4123**	**54.06**	**34.10**	**23.96**	**142.78**	**55.42%**

proposed architecture. The benchmark circuits are synthesized using YOSYS tool using only a library of NAND, NOR, and Inverter gates. The comparison of energy consumption for rCIM vs ASIC implementations is illustrated in Fig. 10. The proposed rCIM architecture saves 89.52% energy while consuming 0.7 ns lower latency in Adder-128 benchmark circuit. The Square-root circuit consumes 90% lower energy with 11 ns lower latency compared to ASIC implementation. The Log2 circuit, which involves a significant number of NAND and NOR operations, exhibits a 100 ns increase in latency, leading to a 13.3% rise in energy consumption for the rCIM implementation. The latency increase is largely due to the multistage combinational circuit design. Given the rCIM architecture's capability to facilitate parallel processing, operations are confined to a single stage to protect data integrity. It is worth noting that this article does not include the data movement time in ASIC implementation, which results in the rCIM displaying a greater latency. The Divisor circuit, with its substantial NOR operation count, achieves 71% reduction in energy consumption compared to its ASIC implementation due to higher power consumption despite having lower latency. Similarly, the Multiplier circuit, with its higher NOR operation count, achieves a 30.8% lower energy consumption despite consuming $12\times$ higher latency, due to a large number of stages in its design. On average, the proposed rCIM macro demonstrates a significant energy savings of 55.42% across all the benchmark circuits.

Table. II compares the proposed rCIM with previous in-memory logic implementations. The proposed rCIM macro operates with an energy efficiency of 65 fJ/NAND operation and 116 fJ/NOR operation. The throughput of the rCIM ranges between 88.2 $GOPS$ while exclusively using NAND operations to 106.6 $GOPS$ while exclusively using NOR operations at 1 GHz. In [10], the input samples are streamed onto the bitlines, which can cause read-disturbs. The proposed rCIM employs dedicated dual read ports that ensure read data stability and achieves $3\times$ higher frequency resulting in $10\times$ higher throughput. In [4], the use of Dual-Split-VDD requires analog expertise and additional engineering effort. The proposed rCIM achieves $2.15\times$ higher throughput compared to [4]. In [22], dedicated read ports ensure data stability but are limited to performing logical operations on either row or column, requiring a sense amplifier for each row and column. The proposed rCIM eliminates single row or column constraints and achieves $1.7\times$ higher throughput while using a single sense amplifier for every two columns. In [17], the transposable 8T cell performs multi-bit "add" and "multiplication" operations but has a lower frequency that results in higher energy/operation consumption. The proposed rCIM achieves $2\times$ higher frequency resulting in $2.6\times$ higher throughput with $1.6\times$ higher energy efficiency compared to [17].

TABLE II: Comparison of rCIM with prior CIM works, illustrating $2.6\times$ higher throughput compared to [17], $2.15\times$ higher throughput compared to [4] and, $1.3\times$ higher energy efficiency compared to [10].

	This work	JSSC'18 [10]	TVLSI'23 [4]	JSSC'21 [22]	ISSCC'19 [17]
Technology	28nm	65nm	28nm	28nm	28nm
Cell Type	10T dual read port	6T	6T	10T	8T
Array Size	256x256	512x256	64x64	64x64	(128x256)x4
Supply Voltage	1V	1V	0.9-1V	0.9V	0.6-1.1V
Frequency	1GHz	NA	0.943-1.18GHz	∼0.300GHz	0.475GHz
Throughput (GOPS)	88.2-106.6	8.26	25.5-41	50	32.7
Energy Efficiency (TOPS/W)	8.64-10.45	6.25	NA	NA	0.55 (mult) 5.27 (add)
Type of Functions	SRAM/LOGIC (NAND, NOR, NOT)	SRAM, AND, MAC	SRAM, AND, NOR, CAM	SRAM, CAM, Logic, Matrix transpose	Logic, Add, Sub, Mult, Div, FP

V. Conclusion

In this paper, we proposed an energy-recycling Resonant Compute-in-Memory (rCIM) architecture to perform boolean logic operations within the SRAM cache memory. Additionally, the use of a series LC resonance-based write driver significantly reduces the dynamic power consumption by recycling

the dissipated energy. The proposed architecture consumes 65 fJ per NAND operation and 116 fJ per NOR operation. Moreover, it saves 55.42% of energy on average compared to traditional von-Neumann architecture while achieving a throughput of 88.2-106.6 $GOPS$ and energy efficiency of 8.64-10.45 $TOPS/W$.

REFERENCES

[1] "Foundations required for novel compute," https://www.darpa.mil/program/foundations-required-for-novel-compute.

[2] C.-J. Jhang, C.-X. Xue, J.-M. Hung, F.-C. Chang, and M.-F. Chang, "Challenges and trends of SRAM-based computing-in-memory for AI edge devices," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 5, pp. 1773–1786, 2021.

[3] Y. Chen, J. Mu, H. Kim, L. Lu, and T. T.-H. Kim, "BP-SCIM: A reconfigurable 8T SRAM macro for bit-parallel searching and computing in-memory," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 5, pp. 2016–2027, 2023.

[4] Y. Wang, S. Zhang, Y. Li, J. Chen, W. Zhao, and Y. Ha, "A reliable and high-speed 6T compute-SRAM design with dual-split-VDD assist and bitline leakage compensation," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 31, no. 5, pp. 684–695, 2023.

[5] N. Rohbani, M. A. Soleimani, and H. Sarbazi-Azad, "PIPF-DRAM: Processing in precharge-free DRAM," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, ser. DAC '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 1075–1080. [Online]. Available: https://doi.org/10.1145/3489517.3530573

[6] H. Cai, Z. Bian, Y. Hou, Y. Zhou, J.-l. Cui, Y. Guo, X. Tian, B. Liu, X. Si, Z. Wang, J. Yang, and W. Shan, "33.4 A 28nm 2Mb STT-MRAM computing-in-memory macro with a refined bit-cell and 22.4 - 41.5TOPS/W for AI inference," in *2023 IEEE International Solid-State Circuits Conference (ISSCC)*, 2023, pp. 500–502.

[7] Y. Li, J. Chen, L. Wang, W. Zhang, Z. Guo, J. Wang, Y. Han, Z. Li, F. Wang, C. Dou, X. Xu, J. Yang, Z. Wang, and D. Shang, "An ADC-less RRAM-based computing-in-memory macro with binary CNN for efficient edge AI," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 70, no. 6, pp. 1871–1875, 2023.

[8] M. Kim, M. Liu, L. R. Everson, and C. H. Kim, "An embedded NAND flash-based compute-in-memory array demonstrated in a standard logic process," *IEEE Journal of Solid-State Circuits*, vol. 57, no. 2, pp. 625–638, 2022.

[9] X. Si, Y.-N. Tu, W.-H. Huang, J.-W. Su, P.-J. Lu, J.-H. Wang, T.-W. Liu, S.-Y. Wu, R. Liu, Y.-C. Chou, Y.-L. Chung, W. Shih, C.-C. Lo, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, N.-C. Lien, W.-C. Shih, Y. He, Q. Li, and M.-F. Chang, "A local computing cell and 6T SRAM-based computing-in-memory macro with 8-b MAC operation for edge AI chips," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 9, pp. 2817–2831, 2021.

[10] S. K. Gonugondla, M. Kang, and N. R. Shanbhag, "A variation-tolerant in-memory machine learning classifier via on-chip training," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 11, pp. 3163–3173, 2018.

[11] M. Ali, A. Jaiswal, S. Kodge, A. Agrawal, I. Chakraborty, and K. Roy, "IMAC: In-memory multi-bit multiplication and ACcumulation in 6T SRAM array," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 8, pp. 2521–2531, 2020.

[12] S. Cheon, K. Lee, and J. Park, "A 2941-TOPS/W charge-domain 10T SRAM compute-in-memory for ternary neural network," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 5, pp. 2085–2097, 2023.

[13] J. Song, X. Tang, X. Qiao, Y. Wang, R. Wang, and R. Huang, "A 28 nm 16 kb bit-scalable charge-domain transpose 6T SRAM in-memory computing macro," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 5, pp. 1835–1845, 2023.

[14] A. Biswas and A. P. Chandrakasan, "Conv-RAM: An energy-efficient SRAM with embedded convolution computation for low-power CNN-based machine learning applications," in *2018 IEEE International Solid - State Circuits Conference - (ISSCC)*, 2018, pp. 488–490.

[15] K. Prasad, A. Biswas, A. Kabra, and J. Mekie, "PIC-RAM: Process-invariant capacitive multiplier based analog in memory computing in 6T SRAM," in *2023 Design, Automation and Test in Europe Conference and Exhibition (DATE)*, 2023, pp. 1–6.

[16] K. Soundrapandiyan, S. K. Vishvakarma, and B. S. Reniwal, "Enabling energy-efficient in-memory computing with robust assist-based reconfigurable sense amplifier in SRAM array," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 13, no. 1, pp. 445–455, 2023.

[17] J. Wang, X. Wang, C. Eckert, A. Subramaniyan, R. Das, D. Blaauw, and D. Sylvester, "14.2 a compute sram with bit-serial integer/floating-point operations for programmable in-memory vector acceleration," in *2019 IEEE International Solid- State Circuits Conference - (ISSCC)*, 2019, pp. 224–226.

[18] J. Chen, W. Zhao, Y. Wang, and Y. Ha, "Analysis and optimization strategies toward reliable and high-speed 6T compute SRAM," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 4, pp. 1520–1531, 2021.

[19] J. Chen, W. Zhao, Y. Wang, Y. Shu, W. Jiang, and Y. Ha, "A reliable 8T SRAM for high-speed searching and logic-in-memory operations," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 30, no. 6, pp. 769–780, 2022.

[20] J. Wang, X. Wang, C. Eckert, A. Subramaniyan, R. Das, D. Blaauw, and D. Sylvester, "A 28-nm compute sram with bit-serial logic/arithmetic operations for programmable in-memory vector computing," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 1, pp. 76–86, 2020.

[21] 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.

[22] Z. Lin, Z. Zhu, H. Zhan, C. Peng, X. Wu, Y. Yao, J. Niu, and J. Chen, "Two-direction in-memory computing based on 10T SRAM with horizontal and vertical decoupled read ports," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 9, pp. 2832–2844, 2021.

[23] 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*, vol. 65, no. 12, pp. 4219–4232, 2018.

[24] Z. Lin, Z. Tong, F. Wang, J. Zhang, Y. Zhao, P. Sun, T. Xu, C. Zhang, X. Li, X. Wu, W. Lu, C. Peng, Q. Zhao, and J. Chen, "In situ storing 8T SRAM-CIM macro for full-array boolean logic and copy operations," *IEEE Journal of Solid-State Circuits*, vol. 58, no. 5, pp. 1472–1486, 2023.

[25] Z. Lin, Q. Jin, Z. Tong, G. Li, S. Gu, J. Li, R. Wang, X. Shu, S. Yu, S. Yan, L. Liu, L. Hao, Q. Zhao, C. Peng, and X. Wu, "Configurable and high-throughput CIM SRAM for boolean logic operation with 321 GOPS/kb and 164395.6 GOPS/mm2," *IEEE Solid-State Circuits Letters*, vol. 6, pp. 153–156, 2023.

[26] R. Islam, B. Saha, and I. Bezzam, "Resonant energy recycling SRAM architecture," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 4, pp. 1383–1387, 2021.

[27] G. Matthew and I. Riadul, "Current-mode clock distribution," 2017, US Patent 9787293B2.

[28] F. u. Rahman and V. Sathe, "Quasi-resonant clocking: Continuous voltage-frequency scalable resonant clocking system for dynamic voltage-frequency scaling systems," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 3, pp. 924–935, 2018.

[29] V. Sathe, "Quasi-resonant clocking: A run-time control approach for true voltage-frequency-scalability," in *2014 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, 2014, pp. 87–92.

[30] D. Challagundla, M. Galib, I. Bezzam, and R. Islam, "Power and skew reduction using resonant energy recycling in 14-nm FinFET clocks," in *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2022, pp. 268–272.

[31] D. Challagundla, I. Bezzam, and R. Islam, "Design automation of series resonance clocking in 14-nm FinFETs," *Circuits, Systems, and Signal Processing*, Aug. 2023. [Online]. Available: https://doi.org/10.1007/s00034-023-02458-4

[32] I. Bezzam, C. Mathiazhagan, T. Raja, and S. Krishnan, "An energy-recovering reconfigurable series resonant clocking scheme for wide frequency operation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 62, no. 7, pp. 1766–1775, 2015.

[33] J. M. Rabaey, *Digital integrated circuits : a design perspective.*, 2nd ed., ser. Prentice Hall electronic and VLSI series. Upper Saddle River, N.J: Pearson Education, 2004.

[34] L. Amarú, P.-E. Gaillardon, and G. De Micheli, "The EPFL combinational benchmark suite," in *Proceedings of the 24th International Workshop on Logic & Synthesis (IWLS)*, no. CONF, 2015.

[35] C. Wolf, "Yosys open synthesis suite," https://yosyshq.net/yosys/.