Ising-CIM: A Reconfigurable and Scalable Compute Within Memory Analog Ising Accelerator for Solving Combinatorial Optimization Problems

Shanshan Xie, *Graduate Student Member, IEEE*, Siddhartha Raman Sundara Raman, *Member, IEEE*, Can Ni, *Member, IEEE*, Meizhi Wang, *Graduate Student Member, IEEE*, Mengtian Yang, and Jaydeep P. Kulkarni, *Senior Member, IEEE*

***Abstract*— Combinatorial optimization problems (COPs) find applications in real-world scientific, industrial, and societal scenarios. Such COPs are computationally NP-hard, and performing an exhaustive brute force search for the optimal solution becomes untenable as the COP size increases. To expedite the COP computation, the Ising model formalism is used, which abstracts spin dynamics in a ferromagnet. The spins are orientated to reach the minimum energy state, representing the optimum COP solution. Previous Ising engine designs utilized dedicated annealing processors or additional digital arithmetic circuits next to the memory bitcells. These custom circuits or processors cannot be repurposed for other applications, incurring significant area and power overhead. In contrast to the prior approaches, this work presents a reconfigurable and scalable compute-within-memory analog approach for Ising computation (called Ising-CIM). This area-efficient approach repurposes existing embedded memory bitcell columns and peripheral circuits to perform analog domain Hamiltonian calculations on the bitlines minimizing area and power overhead significantly. A 13.18-Kb silicon prototype, implemented in a 65-nm CMOS process, demonstrates the Ising-CIM concept and functionality using a 100 × 64 pixel image in a max-cut COP. The Ising-CIM design achieves 48-μm^2/spin unit spin area and 1091× speedup in annealing time compared to the CPU.**

***Index Terms*— Analog computation, compute-in-memory, Hamiltonian, hardware accelerator, Ising model, max-cut problem, simulated annealing.**

I. Introduction

COMBINATORIAL optimization is a branch of mathematical optimization that focuses on finding the optimal or nearly optimal solutions among a finite but extensive collection of possibilities [1]. The combinatorial optimization problems (COPs) find many social and industrial data-intensive computing applications. Examples include optimization of mRNA sequences for COVID-19 vaccines [2], [3], semiconductor supply chains [4], [5], and financial index tracking [6], to name a few. Such COPs are predominantly NP-hard [7] (NP = Non-deterministic Polynomial-time), and performing an exhaustive brute force search becomes untenable as the COP size increases. An efficient way to solve COPs is to let nature perform a potentially exhaustive search in the physical world using an approach based on the Ising model, which can map different types of COPs [8]. The Ising model describes spin dynamics in a ferromagnet [9], wherein spins naturally orient to achieve the lowest ensemble energy state of the Ising model, representing the optimal COP solution [10].

One of the approaches to design Ising hardware is to abstract spin dynamics with a mathematical model (called Hamiltonian) and update spins iteratively to minimize the Hamiltonian. Spin update computation involves numerous memory read and write operations. Consequently, iterative Ising model computing with conventional Von Neumann architectures, as shown in Fig. 1, requires frequent off-chip memory accesses, resulting in degraded performance and high power consumption. For minimizing the off-chip memory accesses, prior approaches have demonstrated compute-near-memory (but not within memory) designs, which performs massively parallel computations near the memory bitcells with dedicated digital arithmetic circuits [7], [11]–[14]. Previous compute-near-memory-based Ising accelerators have reported up to 26 000× speedup [11] and ∼1000× lower energy [13] than CPUs. These custom designs implement multiple instances of digital arithmetic logic (i.e., full adder and XNOR logic) adjacent to unit memory segments, as depicted in Fig. 1. It incurs a significant area overhead, increasing the hardware cost, and can degrade the bitcell density by a factor of 3×–10×. Moreover, when scaling up the spin count, these designs adopt a multi-chip approach [13], which reintroduces the off-chip access latency and energy penalties. Despite impressive performance benefits, the need for dedicated digital arithmetic and off-chip memory access overheads has limited

Manuscript received December 15, 2021; revised March 21, 2022 and May 12, 2022; accepted May 13, 2022. This article was approved by Associate Editor Meng-Fan Chang. This work was supported in part by the NSF CAREER Award, in part by the Intel Rising Star Faculty Award, and in part by the Micron Foundation Faculty Awards. *(Corresponding author: Shanshan Xie.)*

The authors are with the Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX 78712 USA (e-mail: sxie@utexas.edu; s.siddhartharaman@utexas.edu; can5@utexas.edu; wang.mz@utexas.edu; mengtian.yang@utexas.edu; jaydeep@austin.utexas.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSSC.2022.3176610.

Digital Object Identifier 10.1109/JSSC.2022.3176610

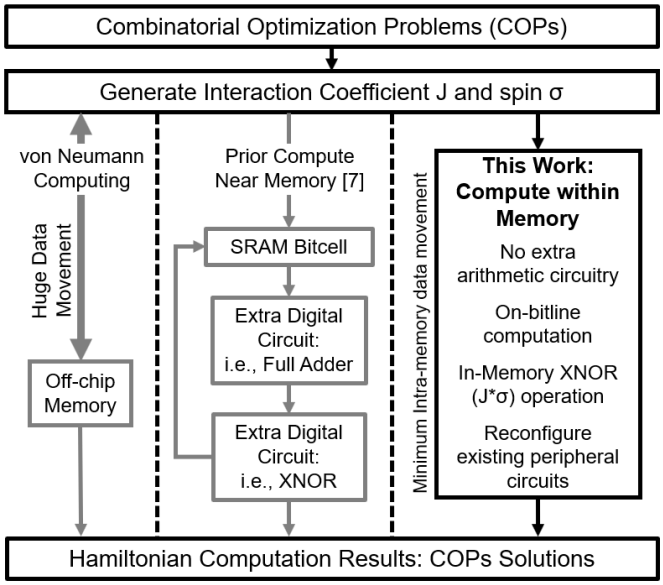


Fig. 1. Comparison of the proposed CIM Ising approach with von-Neumann and near-memory compute methods.

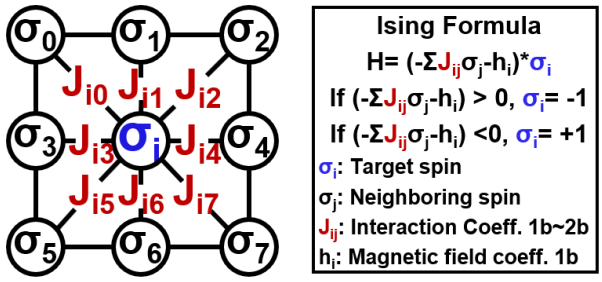


Fig. 2. Ising model formalism.

the adoption of compute-near-memory-based Ising designs in modern SoCs. Therefore, there is a critical need for energy-efficient and cost-effective (area efficient) hardware designs to advance the COP accelerator development.

The concept of compute-in-memory has been widely adopted for deep neural networks [15]. However, in a neural network, the outputs from the previous layer need to be fed into the next layer. Therefore, partial sums or temporary storage resources are needed. In contrast, the Ising computation is naturally more suitable for the compute-in-memory operation because each spin state (which is 1 bit) can be mapped to a physical memory bitcell, and after each Hamiltonian iteration, one-bit spins can be updated locally. This allows the spins to be stored stationary inside the memory array with minimum intra-memory data movement. In addition, the local spin update process reuses available sense amplifiers (SAs) in the memory array and performs a local 1-bit read-modify-write operation to eliminate the need for any analog-to-digital converters (ADCs).

This article proposes a reconfigurable and scalable compute-within-memory (CIM) Ising approach to perform Hamiltonian computations in an analog domain within a memory array with minimal circuit changes. It maps Hamiltonian computations onto available memory wordline (WL) and bitline circuits. The key contributions of this work are as follows.

1) A unique analog domain CIM approach is demonstrated, which mitigates the off-chip data movement between CPU and memory by performing Ising computations directly within the memory array.
2) All Ising computations are performed by reconfiguring the existing peripheral circuitry (i.e., SA and WL drivers) within a memory array without requiring dedicated digital arithmetic circuits. Therefore, the memory arrays can be configured as regular memory arrays used in non-Ising computations and can be reconfigured seamlessly as Ising CIM engines, which significantly eliminates the area overhead due to a dedicated Ising accelerator digital arithmetic circuits.
3) The local write-after-read mechanism is leveraged for updating one-bit spin states, minimizing the intra-memory movement for reading/writing the spins in King's graph. In addition, this approach eliminates the need for a dedicated ADC since the Hamiltonian computation results are used to write back (update) the spin bits.
4) The concept of "ghost cells" is leveraged [16] for seamlessly mapping large Ising models across multiple memory banks, thus enabling scalable Ising-CIM designs for solving complex COPs.
5) The proposed Ising-CIM design supports multi-bit precision for King's graph coefficients (J) by leveraging the available read-assist circuits to generate optimum read WL (RWL) underdrive voltage.
6) 65-nm CMOS silicon prototype measurements demonstrate $6\times\sim17\times$ spin area reduction and $4\times$ annealing time speedup compared to prior hardware approaches. Compared to the CPU annealing time, the Ising-CIM accelerator can improve the annealing time by $1091\times$ as the number of spins increases to 144 K.

This article is organized as follows. Section II introduces the background of this work, including the Ising model and prior approaches. Section III presents the overall Ising-CIM architecture, within-memory XNOR operation, computational dataflow, ghost cell concept, and simulated annealing in the Ising-CIM design. Section IV presents a 65-nm silicon prototype measurement results in detail. Section V concludes this article by highlighting key design attributes of the proposed Ising-CIM approach.

II. Background

A. Ising Model

The Ising model [9] is a mathematical model of ferromagnetism in statistical mechanics [17], which models spin–spin interactions with nearest neighbors, as shown in Fig. 2. The spin (σ) can be one of the two states: $+1$ for up-spin ($\uparrow$) and -1 for down-spin ($\downarrow$). The Ising model can be emulated by leveraging natural coupling phenomena using quantum bits, lasers, or coupled oscillators [18]–[20]. Each one necessitates specialized hardware and a method for converting physical

TABLE I
COMPARISON OF ISING MODEL HARDWARE DESIGNS

	Physical Ising Models			***Hamiltonian based models***					
	D-Wave [18]	**PRL'19 [19]**	**VLSI'20 [25]**	**JSSC'16 [14]**	**ISSCC'19 [11]**	**JSSC'22 [7]**	**JSSC'21 [12]**	**ISSCC'21 [13]**	**This Research**
Technology	Superconductor	Photonics	Coupled oscillators	65nm CMOS	40nm CMOS	65nm CMOS	65nm CMOS	40nm CMOS	65nm CMOS
Operating Temperature	Ultra-Low Temp. (15mK)	Room Temp. (300K)		Room Temp. (300K)					Room Temp. (300K)
Number of spins	128-2K	2K	-	20K	2*30K	480	512	147K	6.4K
Supply (V)	-	-	1V	1.1V	1.1V	0.5-1.2V	1.1V	1.1V~2.5V	1V
Spin Type	Qubit	Optical laser	Ring oscillators	SRAM	SRAM	Register	SRAM	Flip-flop	Embedded memory
Spin Update	Dedicated hardware		Latch-based coupling	Dedicated logic near memory bitcells					Within memory
Graph Model (# Neighbors)	Chimera (6× Spins)	Mobius-Ladder graph	hexagonal structure (Max neighbors)	2× 2D Lattice (5× Spins)	King's graph (8× Spins)	King's graph (8× Spins)	Fully Connected	Sparse king's graph	King's graph (8× Spins)
Spin Area (μm^2)	13,888	-	-	289	339	832	-	552	48

state information to digital bits. Alternatively, Ising spin dynamics can be simulated as a computational data flow realized in CMOS technologies. In this case, Ising spin dynamics are represented mathematically in terms of a spin interaction coefficient (J) and an external magnetic field (h_i) affecting the energy state of all individual spins (H_σ)

$$H = -\sum_{ij} J_{ij}\sigma_i\sigma_j - \sum_i h_i\sigma_i. \tag{1}$$

The computing steps involve minimizing the Hamiltonian (H) energy function expressed in (1), where σ_i represents the target spin, σ_j is the neighboring spin, and J_{ij} is the interaction coefficient. Here, i and j correspond to spin indices of a representative 3×3 matrix connected as King's graph [21] (see Fig. 2), and the target spin σ_i interacts with its eight neighboring spins $\sigma_{0\sim7}$

$$H_\sigma = -\sum_j J_{ij} * \sigma_j - h_i. \tag{2}$$

The Ising model-based Hamiltonian computation to reach the minimum energy state is performed in two steps: 1) local spin update based on the neighboring spins and 2) annealing process to avoid being stuck in local minima. In (2), H_σ is defined to represent the local energy for a given target spin. If H_σ exceeds a certain threshold (0 in Fig. 2), then the spin state (σ_i) is updated to -1. Otherwise, it is updated to $+1$, as shown in (3). This iterative process is the first step of minimizing the energy state, called local spin update [7], [22]

$$\sigma_i = \begin{cases} -1, & \text{if } H_\sigma > 0 \\ +1, & H_\sigma < 0 \\ +1/-1, & H_\sigma = 0. \end{cases} \tag{3}$$

B. Prior Work

Ising model hardware designs aim to achieve the optimal solution for a given COP by emulating Ising spin dynamics using either: 1) physical systems that tend to evolve toward their lowest energy state or 2) by mathematically abstracting spin dynamics as a Hamiltonian function and solving it iteratively.

1) Physical Ising Models: Table I shows the previous implementations of such physical systems, including quantum, photonic, and coupled oscillators. D-wave, a quantum annealing-based quantum computer, is presented in [18], which utilizes quantum bits (qubits) to encode one, zero, or both information simultaneously. However, the design necessitates a cryogenic operating temperature (∼15 mK), which results in substantial cooling power costs (25 kW) [7]. The optical Ising machine, on the other hand, embeds spin states into a binary phase modulator of an optical field. With spin couplings set by input amplitude modulation and a feedback scheme, global minima of the spin energy can be determined using light propagation [19]. However, large size optical components and special fabrication requirements may limit the scaling of the photonic Ising approach to large COPs.

In contrast to the above two approaches, CMOS coupled oscillator design has the advantages of room temperature operation and miniaturization without compromising accuracy. The coupled oscillator dynamics settle to a steady state, and the phase of coupled oscillators determines the ground state of the Ising model [20], [23]–[25]. However, coupled oscillator approaches consume significant dynamic power due to the constant toggling of the oscillator nodes. Furthermore, these physical systems are susceptible to process variations, and unintentional spurious coupling events can cause ground state fluctuations. As a result, scaling to large COPs can be a challenge for physical systems emulating Ising spin dynamics.

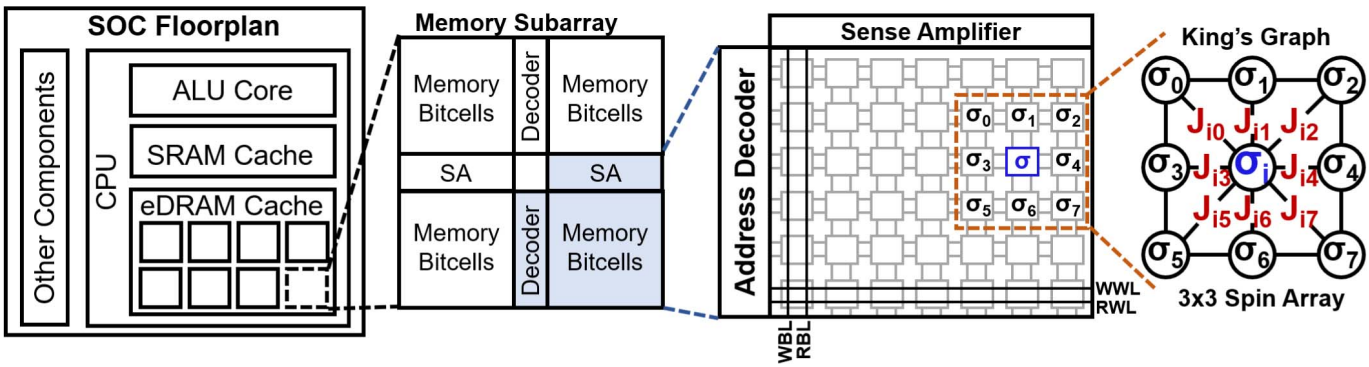


Fig. 3. SoC floorplan, Ising-CIM architecture, and King's graph mapping to perform the accurate Ising CIM computations.

2) Iterative Ising Models: CMOS Ising designs map the COP into an Ising model and solve iteratively (1), which are easy to design, manufacture, and scale. Yamaoka *et al.* [14] presented a CMOS annealing implementation that uses the interaction of connected spins to move toward the lowest energy level. Dedicated arithmetic circuits (i.e., XORs, switches, and majority voting circuits) are employed along with SRAM bitcells.

Recently, a digital annealing processor (AP) [7] has been presented based on the compute-in(near)-memory spin operator and register-based spins to increase energy efficiency and reduce annealing time. This design eliminates the need for static random access memory (SRAM) read/write operations to access the adjacent spin values and mitigates the need for a local SA triggering to read coefficients from SRAMs. However, this approach adds custom digital arithmetic circuits (four transmission-gate-based XNORs and seven full-adders) in each column along with four SRAM bitcells, and the Hamiltonian computation needs to wait for the partial sum to propagate.

Another COP approach implements a fully connected annealer based on the stochastic cellular automata algorithm [12]. Although some promising innovations are shown (i.e., delta-driven simultaneous spin update and all-to-all connected interactions), this digital annealer design implements dedicated digital arithmetic logic resulting in a compute-near-memory design, which would degrade the memory bitcell density. In addition, in a compute-near-memory design, the energy for moving the data from SRAM to the annealer logic still remains a challenge. Similarly, a 144-Kb AP chip was demonstrated in another recent approach [13], which implements a 9 $\times$ 16k spin system using flip-flop-based structures to connect multiple AP chips through the inter-chip interface. Each AP chip consists of an Ising core, an inter-chip interface, and a controller. These custom arithmetic circuits and dedicated processors incur area overhead and degrade the memory density, thus increasing hardware costs. Moreover, these extra arithmetic circuits are not utilized during non-COP computations, making it costly to adopt compute-near-memory approaches in modern, area-optimized, and high-density memory arrays.

In contrast to prior approaches, this work explores a reconfigurable and scalable CIM design by repurposing the existing embedded memory array and peripheral circuitry for performing Ising Hamiltonian computations.

III. Proposed Ising-CIM Approach

A. Architecture Overview

This section presents a unique CIM design that fundamentally avoids extra digital arithmetic circuits and utilizes the inherent features and structures within a memory array to perform Ising model computations. Fig. 3 shows the big picture of the proposed Ising-CIM accelerator architecture in an SoC design. It is realized by utilizing the baseline cache memory by reconfiguring the available bitcell columns to perform Ising model computations, such as Hamiltonian, spin update, and annealing process. A part of the embedded memory can be configured as an Ising-model engine. On the other hand, the remaining portion of the memory array can be utilized for non-Ising workloads as standard cache memory. Reconfiguring the existing memory array for Ising model computations avoids the area overhead of custom Ising circuits.

Fig. 5 shows the Hamiltonian computation mechanism in the analog domain using either an SRAM bitcell or an embedded Dynamic RAM (eDRAM) bitcell having a dedicated read port. Spins (σ_s) and J coefficients are initially stored in the embedded memory bitcells. During Ising computation, J coefficients are read and applied to the RWLs of the spin array. The mapping between a J coefficient and analog RWL voltage is listed in Table II. $J = -1$ is mapped to V_{SS}, and $J = +1$ is mapped to V_{RWL}, where V_{RWL} is an optimized underdrive voltage ($<V_{DD}$). The underdrive voltage V_{RWL} is selected so that $H_\sigma > 0$ or $H_\sigma < 0$ (3) can be differentiated using an SA. When the V_{RWL} voltage is too low (e.g., 400-mV RWL case in Fig. 4), the RBL discharge rate is reduced, and thus, the RBL voltages for different H_σ's are overlapped, as shown in the Monte Carlo simulation (see Fig. 4). In this case, local spin update cannot be performed accurately since the SA cannot distinguish whether H_σ is larger or smaller than zero. When V_{RWL} voltage is closer to V_{DD}, although the RBL voltages are saturated to V_{SS} for negative H_σ values,

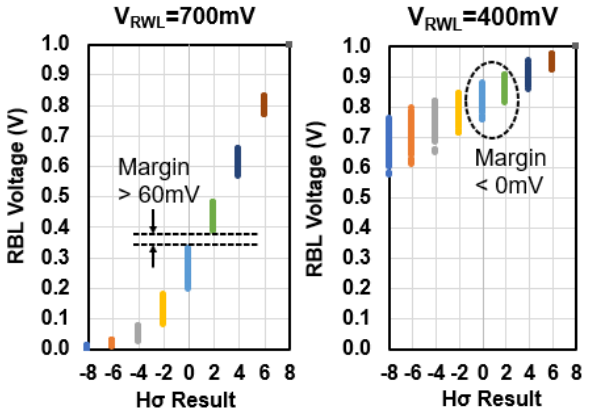


Fig. 4. Monte Carlo simulation of RBL voltages distribution for different H_σ values $\{-8, -6, -4, -2, 0, 2, 4, 6, 8\}$ when $V_{\text{RWL}} = 700$ and 400 mV at $V_{\text{DD}} = 1$ V under cold (-40 °C) temperature.

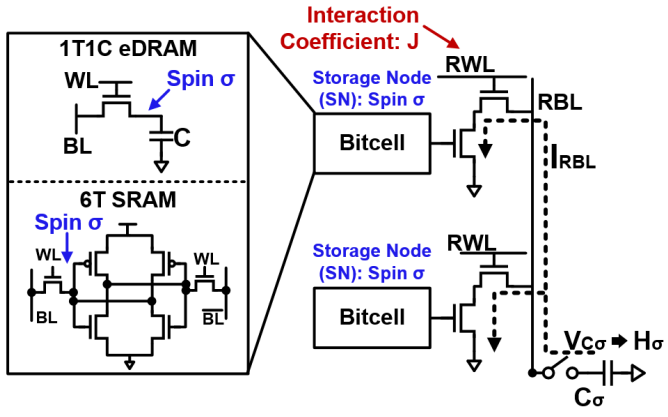


Fig. 5. Hamiltonian analog computation mechanism.

TABLE II

SPIN σ AND J COEFFICIENT MAPPING TO AN ANALOG VOLTAGE

		Bitcell Voltage			RWL Voltage
Spin (σ)	-1	V_{SS} (L)	J Coeff.	-1	V_{SS} (L)
	+1	V_{DD} (H)		+1	V_{RWL} (H)

TABLE III

IN-MEMORY XNOR ($J*\sigma$) ISING OPERATION

σ	J	$\overline{\sigma}$	$\overline{J}$	J*σ	I_{RBL}?	$V_{C\sigma}$
-1	-1	+1	+1	+1	Yes	Discharge
-1	+1	+1	-1	-1	No	Preserve
+1	-1	-1	+1	-1	No	Preserve
+1	+1	-1	-1	+1	Yes	Discharge

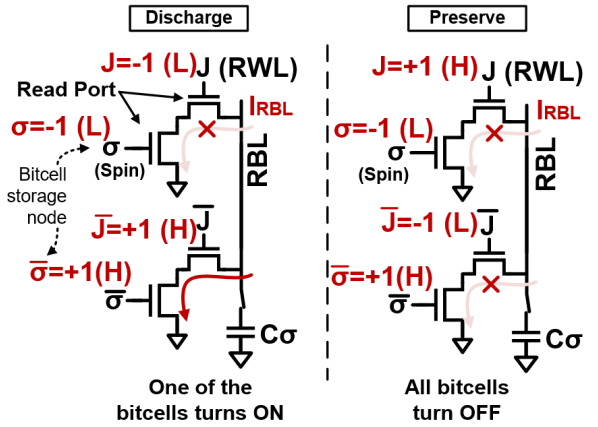


Fig. 6. In-memory XNOR operation example for read bitline discharge and preserve cases.

the voltage difference between maximum V_{RBL} ($@H_\sigma = 0$) and minimum V_{RBL} ($@H_\sigma = 2$) is larger, leading to increased sensing margin for local spin update. This also shows the benefit of using analog computation for local spin update since it only requires one reference comparison (3), instead of accurately distinguishing each H_σ value from -8 to $+8$, at the same time. The spin state (σ) of -1 (down-spin) is mapped to bitcell node value of V_{SS}, whereas spin state of $+1$ (up-spin) is mapped to bitcell value of V_{DD}. The unit capacitor (C_σ) in Fig. 5 is used to perform charge domain H_σ analog computation directly on a bitline. This computation is achieved by repurposing the available bitcell columns in the normal mode of operation (non-CIM) as a charge domain circuit in the Ising-CIM mode by activating multiple WLs simultaneously to discharge the capacitor C_σ. The discharge current reflects the multiplication between a spin state and a J coefficient. The charge sharing among different RBLs reflects the summation in the H_σ equation.

B. In-Memory XNOR Ising Operation

The main operation in (2) is the multiplication between a spin state σ and a J coefficient of King's graph, which is an XNOR operation between σ and J. Table III shows the in-memory XNOR concept by storing both σ and $\overline{\sigma}$ inside the memory array and applying J and $\overline{J}$ on different RWLs. When σ and J are both LOW or both HIGH, the expected result for the multiplication is $+1$. In this case, the read port of one of the bitcells discharges the capacitor C_σ, as shown in Fig. 6. On the other hand, when σ and J have different values, the $J \times \sigma$ result is -1. Therefore, none of the bitcell read ports is turned on to discharge the capacitor C_σ because either the RWL or the bitcell storage node is LOW to preserve the bitline voltage.

C. Ising-CIM Hamiltonian Computation

The Ising-CIM computational data flow consists of Hamiltonian computation, spin update, and annealing process to reach the global minima. It is computed in five steps, as illustrated in Fig. 7. Each step is described as follows.

Step 1 (Precharge): All RBLs are precharged to the supply voltage (V_{DD}) through PMOS transistors with an active-low PRECHARGE signal. This step ensures that all parallel bitline computations start from the same voltage level (V_{DD}).

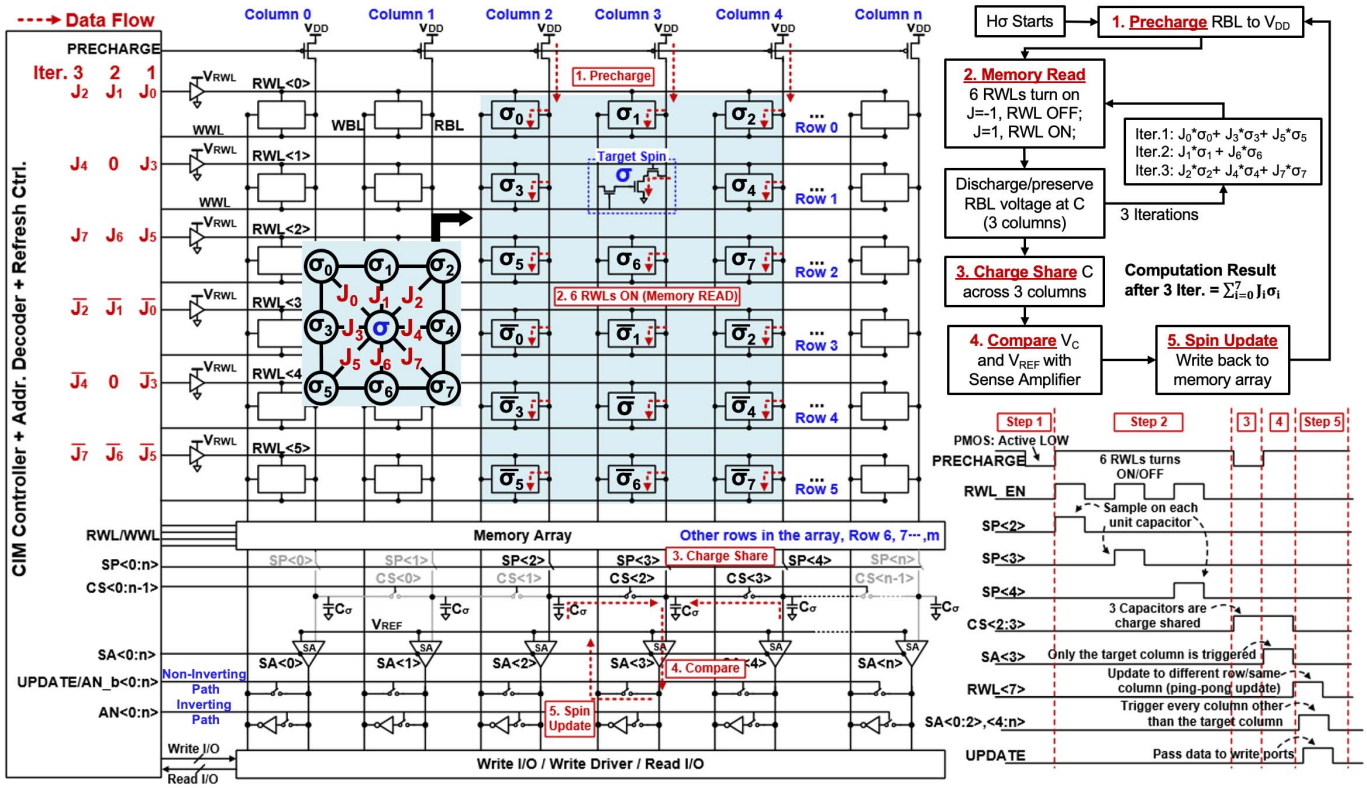


Fig. 7. Overall Ising-CIM circuit schematics, data flow, and timing diagram for performing H_σ computation and spin update.

Step 2 (Memory Read for $-J_i * \sigma_i$ Computation): Step 2 in Fig. 7 illustrates $-J_i * \sigma_i$ computation for a target spin σ. J coefficients associated with the neighboring spins are first read from other section of memory array and then applied onto RWLs by reconfiguring the RWL address decoder. In the first iteration, J_0, J_3, J_5, and their complement values ($\overline{J_0}$, $\overline{J_3}$, and $\overline{J_5}$) are applied onto six RWLs through the WL controller module to perform the multiplication with the second column (σ_0, σ_3, σ_5, and their complementary values). The HIGH (LOW) value of the J coefficients turns on (off) the RWL. If the RWL turns on, the read port of the bitcell is turned on/off depending on the bitcell storage node value, the spin state (σ). This achieves the in-memory XNOR operation for the Hamiltonian computation. Since J coefficients are applied to RWLs and RWLs, and drive the entire rows in the array, SP$\langle 0:n\rangle$ (SP = sample) is used to ensure that the bitline voltage is only sampled to its respective sampling capacitor (C_σ) on the second column. In contrast, sampling capacitors from other columns remain disconnected from the other RBLs. In the second iteration, J_1, J_6, and their complementary values ($\overline{J_1}$ and $\overline{J_6}$) are applied to the RWL controller to perform the multiplication with the spins in column 3. However, in this iteration, row 1 and row 4 (see Fig. 7) are always OFF because the corresponding bitcells in these rows store the target spin (or its complement value), which is not part of its Hamiltonian computation. In other words, only the neighboring eight spins in King's graph are part of the Hamiltonian computation, as shown in Fig. 2 and (2). The third iteration follows a similar step, where J_2, J_4, J_7, and their complement values ($\overline{J_2}$, $\overline{J_4}$, and $\overline{J_7}$) are applied on RWLs, and the multiplication results with σ_2, σ_4, σ_7, and their complementary values are sampled on the fourth column C_σ capacitor. As shown in the timing diagram in Fig. 7, RWL_EN turns on three times to perform the above three iterations, and SP$\langle 2\rangle$ $\sim$ SP$\langle 4\rangle$ turns on one by one for discharging the corresponding C_σ. Multiple such computations can be performed in parallel when J coefficients are the same for the target spins on the same row. In addition, for more complex COPs, parallelism can be achieved by arranging the memory array into sub-arrays and performing parallel computation on each sub-array.

Step 3 (Hamiltonian Computation Using Charge-Sharing): In the illustration shown in Fig. 7, $CS\langle 2\rangle$ and $CS\langle 3\rangle$ turn on to activate the charge share operation among the three sampling capacitors (C_σs) in columns 2–4, performing the summation for eight neighboring spin and J coefficient pairs' multiplication, as expressed in the Hamiltonian equation (2). By using a bus signal ($CS\langle 0:n-1\rangle$) to control the charge share operation, C_σ's on the other columns are isolated for a specific King's graph calculation.

Step 4 (Spin-Flip Threshold Comparison): The existing SA in the peripheral memory circuit is utilized. By asserting SA$\langle 3\rangle$, the voltage (V_{C_σ}) at C_σ in column 3 is compared with a reference voltage V_{REF} ($V_{\mathrm{REF}} = V_{\mathrm{DD}}/2$). One of the available memory columns can be used to generate the V_{REF} internally by turning on multiple RWLs for a certain pre-defined pulsewidth for discharging the RBL to the target reference voltage. Fig. 8 shows the circuit diagram, simulation waveform to illustrate the concept, and Monte Carlo simulations for V_{REF}. In addition, the number of turned on RWLs, RWL pulsewidth, and WL underdrive voltage can be tuned for calibrating V_{REF}.

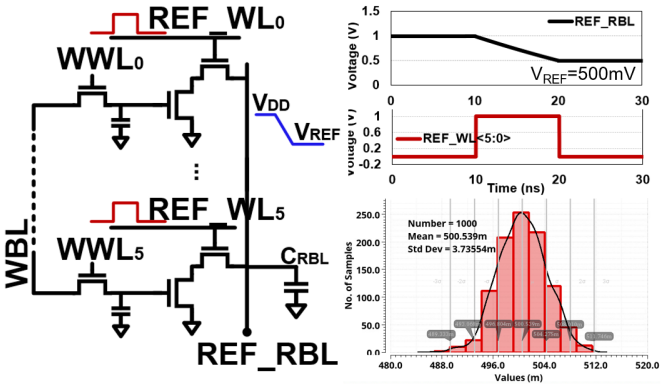


Fig. 8. Reference voltage (V_{REF}) generation.

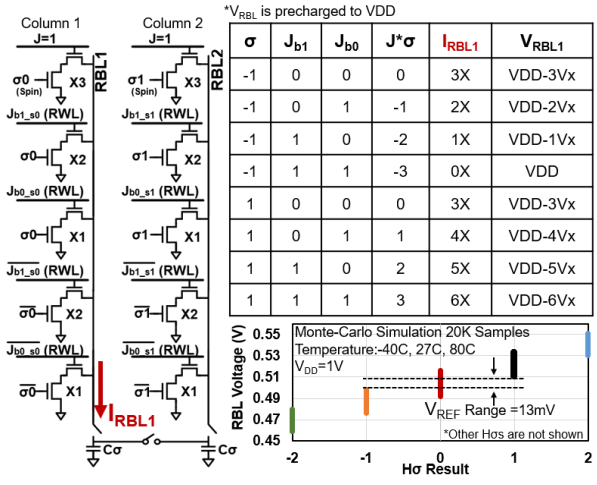


Fig. 9. Multi-bit precision J coefficient Hamiltonian computation circuit diagram, operation truth table, and Monte Carlo simulation.

For calibration, a known sequence of spins needs to be fed into the array to monitor the SA output. In this case, a dedicated V_{REF} generation circuit can be avoided, which further reduces the area overhead and hardware implementation cost.

Step 5 (Spin-State Update): The updated spin state is stored in another bitcell row without disturbing the original target spin in row 1. This is because the initial value of the target spin is still required, as it is a part of King's graphs of its neighboring spins and would be used for computations of H_σ of respective neighboring spins. In this case, RWL⟨7⟩ is turned on to read the row-7 bitcell values on the RBLs. After that, all SAs turn on, except for column-3, to not disturb the computation result at the output of column 3 SA. At the same time, WWL⟨7⟩ and UPDATE signals are also turned on to pass the write port data to bitcells, including the updated spin value.

D. Multi-Bit Precision J Coefficients

For multi-bit J coefficient computation, the data flow is similar to the single bit J coefficient computation. Initially, the bitlines are precharged to V_{DD}, and the spins and their complement values are stored in the bitcells, as shown in Fig. 9. The first six bitcells in the column are storing the spin, and

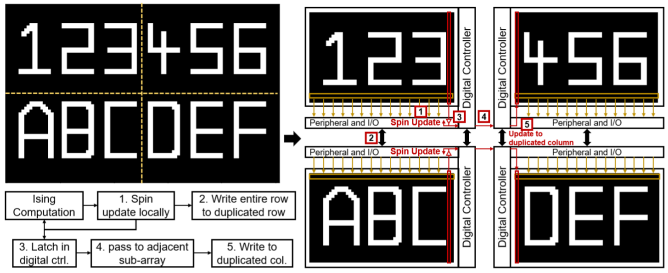
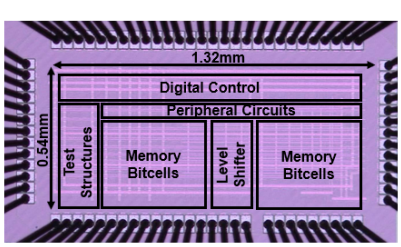


Fig. 10. Ghost cell concept.

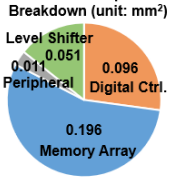
the bottom three bitcells are storing its complementary value ("X3" means that three bitcells are used). Each J coefficient bit is fed onto the corresponding RWL, and the J coefficient for the first three bitcells is always "1" to generate a constant three unit current offset. When spin equals "+1," the discharge current varies from 3× to 6× depending on the J coefficients. In this case, the RBL is discharged to a voltage ranging from V_{DD}-3V_X to V_{DD}-6V_X, as shown in the table in Fig. 9, where V_X is a unit discharge voltage. After each bitline voltage is established, charge-share and the similar steps described in Section III-C are performed for the Hamiltonian computation. Compared with single bit J coefficient computation, spin values are stored across the column without sharing the same bitline. Therefore, eight computations are required for the eight neighboring spins. For performing multi-bit J computation, the WL underdrive voltage needs to be optimized for the SA to distinguish between $H_\sigma \leq 0$ and $H_\sigma > 0$. Monte Carlo simulations are performed to verify the RBL voltage for each H_σ case under local variations, and for illustration purpose, other H_σ results are not shown in Fig. 9.

E. Ising-CIM Scalability

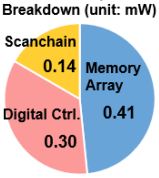
As the size of COP increases, the size of the Ising model would scale up, which would necessitate a large capacity memory bank to map the entire King's graph. However, memory banks in current CMOS technologies are typically built to optimize WL and BL interconnect delays and have limited bit capacity. This scalability aspect of the Ising model is addressed by leveraging the concept of "ghost cells" [16] (see Fig. 10), which splits a large King's graph into multiple sub-King's graphs and maps the sub-graphs across various memory banks for Ising-CIM computation. The bits (spins) of a sub-King's graph, which are at the edge of a given memory bank, are duplicated in adjacent banks, and as the memory bank size increases, the "ghost cells" overhead reduces. The spins on these duplicated horizontal and vertical edges need to be synchronous for the next Ising computation. Therefore, for the horizontal duplicated row, the entire row data are passed to the adjacent sub-array after all the spin update processes are completed. For the vertical duplicated column, after each spin update cycle for the edge spin, the updated spin is latched to the adjacent column through the digital controllers. In order to update the spin in a specific location in a column, a read-modify-write step needs to be performed for the entire row. In this case, the write data for the edge column are overwritten by the new spin values, and the refresh operation is performed on the other columns.



65nm Test-chip Area Breakdown (unit: mm²)



65nm Test-chip Power Breakdown (unit: mW)



Test-chip Summary	
Design Parameters	
Technology	65nm CMOS
Computing Mechanism	Within-Memory Computing
Core Voltage (V)	0.9 ~ 1.2
Chip Size (mm²)	1.53
[2]Core Size (mm²)	0.71
Bitcell Topology	3T1C Gain-Cell
Bitcell storage cap. (fF)	13
eDRAM Capacitor Type	Metal-oxide-metal (MOM)
Total Memory Capacity	13.18Kb
Spin Area (μm²)	48
Temperature	27C to 60C
Hamiltonian Computation Details	
Annealing Method	Simulated Annealing
Graph Model (# Neighbors)	King's graph (8x Spins)
# of Spins	6.4K
Annealing Time (μs)	51.2 (6.4K)
Measured Results	
[1]Power/spin (μW/spin)	0.11 (@VDD=1V)
[1]Energy/spin (fJ/spin)	13.3 (@VDD=1V)
Single Spin Update Cycle	3: Computation, 1: Spin update

[1]Exclude scanchain power; [2]Exclude padring area;

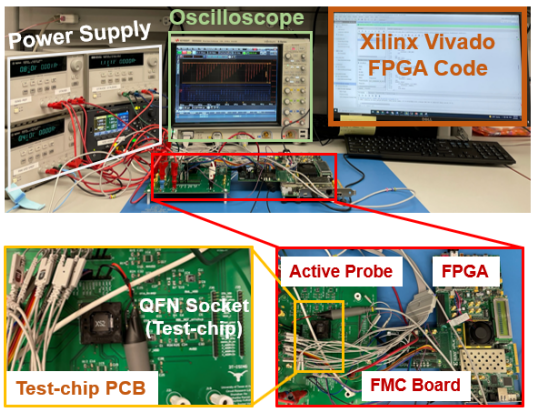


Fig. 11. Ising-CIM silicon prototype die micrograph, area/power breakdown, test-chip summary, and FPGA-based lab experiment setup.

F. Annealing Process

To avoid the Hamiltonian energy function getting stuck in local minima values, an annealing mechanism is adopted by randomly flipping the spin states according to temperature-dependent scheduling [7], thus changing the Hamiltonian energy state. After the local spin update process is completed for all the spins, a random number sequence with 1s and 0s is fed into the AN$\langle 0:n \rangle$ port (shown in Fig. 7), which can be generated externally by specifying the number of spins (N) to be randomly flipped. The random sequence turns on the inverting path when the number is 1 to flip the read bit and turns on the non-inverting path when the number is 0 to preserve the read bit. A regular read operation is activated for the annealing process by precharging the RBL, turning on the RWL, and firing all the SAs. Once the SA outputs are resolved, the randomly flipped spin states are updated with a write-back operation depending on the respective annealing enable bits (AN and AN_b). The number N is controlled externally and is decreased according to the annealing schedule [7], [22]. In the annealing scheduling, a linear (or exponential) cooling schedule is defined, and the temperature (from the algorithm perspective) decreases in each Ising iteration. The acceptance of a new positive energy state depends on the comparison between a random number generated uniformly in the interval [0, 1] [26] and the cooling schedule. The process is repeated until convergence is achieved, and the optimized solution is found. Prior approaches [7], [11], [14], [22] have implemented varied annealing mechanisms depending on the hardware design. Therefore, in order to keep the annealing process flexible and user-controlled, the randomness component and the random bit-streams are controlled off-chip with a scan–chain interface and FPGA in the proposed Ising-CIM design. In this case, different annealing approaches can be easily implemented, customized, and defined by users.

IV. Measurement Results

A. Test-Chip Measurement Setup

Fig. 11 shows the 65-nm CMOS silicon prototype die micrograph, measurement setup, and test-chip summary of the proposed Ising-CIM design. The test chip implements the CIM operations to compute the Hamiltonian, annealing process, and spin updates with King's graph model. The spin states are updated with a write-after-read step, which intrinsically performs a refresh operation for the 3T1C (T = transistor, C = capacitor) eDRAM bitcell used in this memory macro. This relaxes the need for regular refresh operations during Hamiltonian computations, reducing the eDRAM refresh power overhead. As long as the eDRAM refresh interval is longer than the Hamiltonian iteration duration, Ising operations are not required to be paused for a refresh since the refresh operation is inherently and repeatedly triggered during Hamilonian spin-state update steps. The bitcell density improves with compact 3T1C eDRAM bitcells, which can support larger size Ising models for the same memory macro area. It is worth noting that the proposed Ising-CIM approach also can be realized using 6T (with optimized WL underdrive read assist techniques [27]) and/or 8T SRAM bitcells.

The overall measurement setup, test-chip interface, and test methodology consist of: 1) mapping a COP onto King's graph with J coefficients using Python [28] framework, as discussed later in Section IV-B6; 2) feeding the generated J coefficients to the test chip using Xilinx Virtex-7 FPGA VC707 Evaluation board [29] and Xilinx FMC XM105 debug card [30]; 3) oscilloscope demonstration of the Ising-CIM functionality, as shown in Fig. 11; 4) performing on-chip H_σ computation, including spin updates and annealing process; 5) Hamiltonian energy calculation by reading the spins from the memory macro; 6) performing statistical characterization to quantify die-to-die variations on COP accuracy, and annealing time metrics; and 7) power analysis of the Ising-CIM design, and in Fig. 11, the power number includes the data movement from IO pins to the digital controller and the RWLs of the memory array. The cost-effective 65-nm test chip implements 100×64 spins supporting a 100×64 pixel image to evaluate a max-cut COP. A larger COP can be realized either using a large capacity memory macro or using the "ghost cell" concept, which is discussed in Section III-E, by splitting the image and storing the image segments into multiple memory banks.

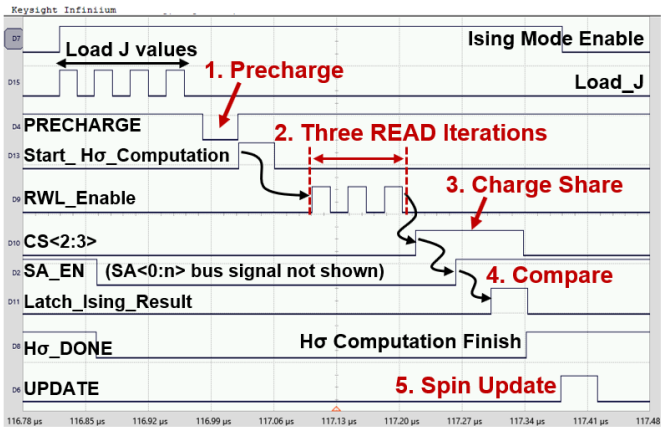


Fig. 12. Oscilloscope demonstration of Ising-CIM H_σ computational data flow. Note that this clock frequency is not the maximum operating clock frequency of the memory array.

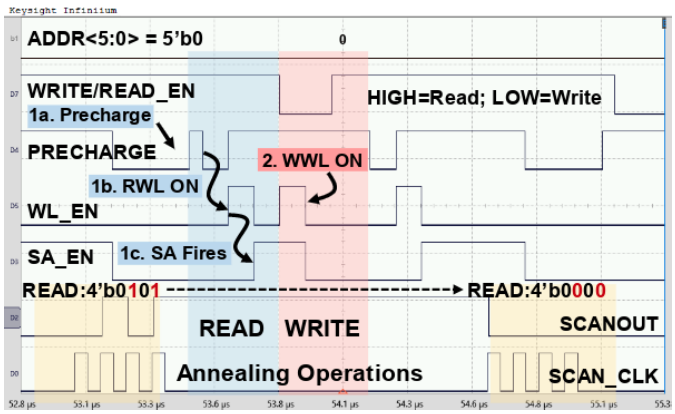


Fig. 13. Oscilloscope demonstration of the simulated annealing functionality by flipping 4'b0101 to 4'b0000.

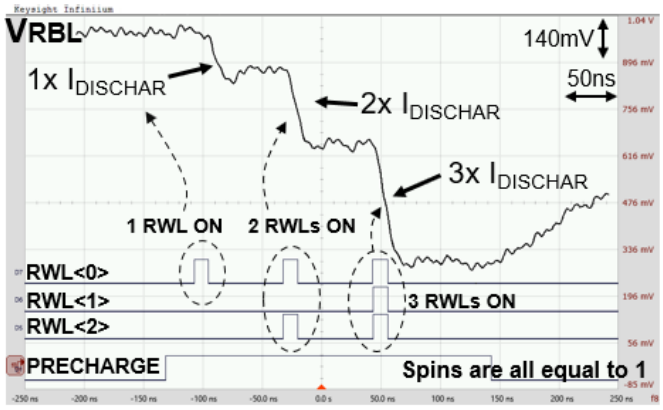


Fig. 14. Oscilloscope demonstration of RBL discharge for different discharge currents performing analog charge domain Hamiltonian computations on the bitline.

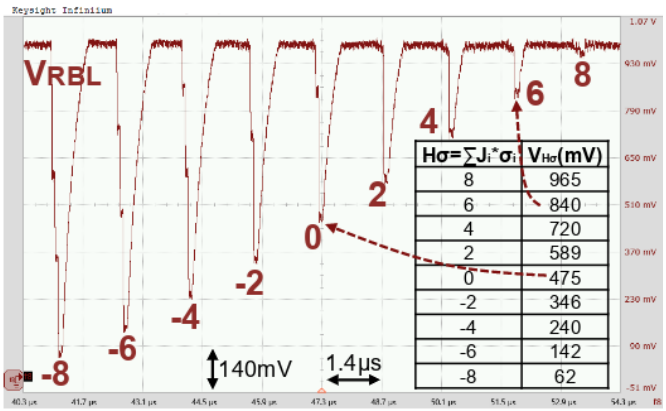


Fig. 15. Oscilloscope demonstration of various H_σ computation results (-8 to $+8$). Note that the clock frequency is reduced only for capturing the analog output waveform.

B. Test-Chip Characterization

1) Ising-CIM H_σ Computation: Fig. 12 shows the oscilloscope waveforms demonstrating the Hamiltonian computation dataflow described earlier in section III-C. Initially, the digital controller reads eight J coefficients of King's graph by asserting the Load_J signal four times. The rest of the signals indicates successful H_σ computation dataflow. It should be noted that the operating frequency is intentionally lowered for the functionality demonstration with oscilloscope waveforms. It is not the maximum operating clock frequency of the memory array.

2) Simulated Annealing Demonstration: Fig. 13 shows the oscilloscope demonstration of the simulated annealing steps. Initially, the read data from the first row are 0101, and the AN_EN⟨3:0⟩ bits are set to 0101. After the simulated annealing process, which is discussed in Section III-F, the data that are read from the bitcell changed from 0101 to 0000. It is worth noting that 4-bit simulated annealing waveforms shown in Fig. 13 are for demonstration purposes, and it is not the maximum memory I/O bandwidth for the test chip. A scan–chain interface is implemented to perform the memory read/write operations, as shown by SCAN_CLK and SCANOUT signals in Fig. 13.

3) H_σ Computation Demonstration on RBL: Fig. 14 shows the oscilloscope demonstration of H_σ on-bitline multiplication as the RBL discharge rate for three different cases, assuming that the spin states are all 1s. The bitline discharge current rate is reflected on the capacitor (C_σ) voltage. Fig. 15 shows various H_σ analog voltage from -8 to $+8$, which offers good linearity across the entire range of J and σ multiplications of a 3×3 King's graph.

4) Process Variation Effects on H_σ Computation: As the spin update is determined by comparing V_{C_σ} with V_{REF}, the PVT variations can introduce uncertainty at the reference boundary when the difference between V_{C_σ} and V_{REF} is within the SA offset voltage. This can result in spin update uncertainty, which may impact the number of annealing steps to reach the global minima or result in a suboptimal COP solution. To evaluate the impact of the process, temperature, and voltage variations, Monte Carlo simulation, as shown in Fig. 16, is performed over 20 K samples among different temperatures (-40 °C, 27 °C, and 80 °C) and different V_{DD} levels (0.9, 1, and 1.2 V). In the Monte Carlo simulation, although there are local variations on RBL voltage for each H_σ, the margin to differentiate $H_\sigma \leq 0$ or >0 is large enough (>60 mV) for the SA, leading to zero SA uncertainty in

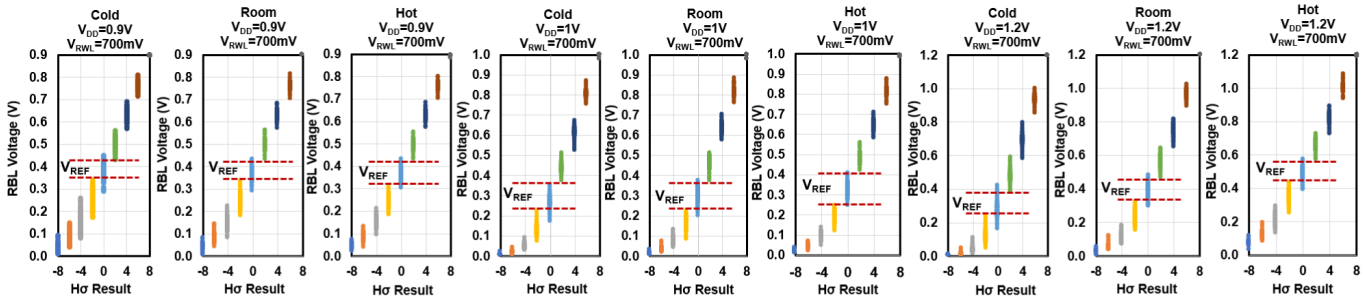


Fig. 16. Monte Carlo simulation for 20 K samples under cold (−40 °C), room (27 °C), and hot (80 °C) temperatures and under three different supply voltages (0.9, 1, and 1.2 V) with the reference voltage range.

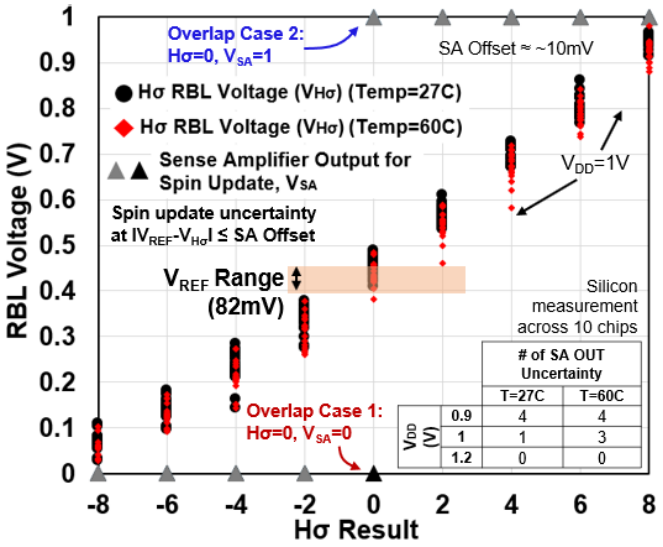


Fig. 17. RBL voltage variations with H_σ comparison results at V_{DD} = 1 V and the SA uncertainty cases when $H_\sigma \approx 0$ when V_{DD} = 0.9, 1, and 1.2 V.

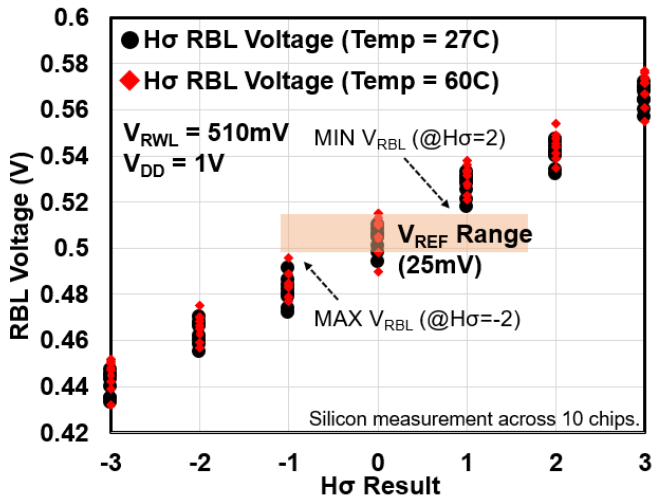


Fig. 18. Measured RBL voltage die-to-die variation under room (27 °C) and hot (60 °C) temperatures for H_σ computation with 2-bit J coefficient.

this case. In addition, measurements were performed on ten test chips at room temperature (27 °C) and at hot temperature (60 °C) with V_{DD} equal to 0.9, 1, and 1.2 V. The measurement data, as demonstrated in Fig. 17 (V_{DD} = 1 V), show that spin update uncertainty happens only when H_σ is around 0. As shown in the table in Fig. 17, this SA output uncertainty occurs in 12 out of 540 measured data points across different temperatures and supply voltage levels. The effect of SA uncertainty and supply voltage on the annealing cycle is shown in Fig. 19.

5) Multi-Bit Precision J Coefficients: The Ising-CIM design supports multi-bit J coefficients with optimized RWL voltage levels generated using the WL underdrive circuit. Fig. 18 shows the measured RBL voltage die-to-die variations for H_σ from −3 to +3. For simplicity, other H_σ results are not shown since the SA only needs to compare near when H_σ equals zero for the local spin update step. The sensing margin is about 20 mV for each die for the local spin update. After calibration, the two cases (3) in the spin update can be clearly detected by the SA leading to zero uncertainty.

6) Max-Cut Problem Demonstration: For a max-cut problem, a 100 × 64 pixel image "123456ABCDEF" is used for quantifying the Hamiltonian energy and annealing time. Initially, the spin values are randomly generated with (100, 64) array size. J coefficients are mapped to −1 when the edges cross the interaction [red connection in Fig. 19(a)] in King's

TABLE IV

ISING-CIM PERFORMANCE COMPARISON WITH PRIOR WORKS FOR HAMILTONIAN COMPUTATION ON CMOS AP

	This work	JSSC'16 [14]	JSSC'22 [7]	ISSCC'21 [13]
CMOS Technology	65nm	65nm	65nm	40nm
Bitcell Topology for Spin Storage	3T1C eDRAM	6T SRAM	Register	Flip flop
Hamiltonian Computation	Within-Memory	Near-Memory	Near-Memory	Near-Memory
Extra Arithmetic Circuitry	No	Yes	Yes	Yes
Annealing Method	Simulated annealing	Simulated annealing	Simulated annealing	Metropolis SA
J Coeff. Bit Width (bit)	1 ~ [2]4	2	4	5
Graph Model (# Neighbors)	King's graph (8x Spins)	2x 2D Lattice (5x Spins)	King's graph (8x Spins)	King's graph (8x Spins)
Operating Voltage (V)	0.9~1.2	1.1	0.5 -1.2	1.1
Single Spin Update Cycles	3 Cycle (Computation) 1 Cycle (Spin Update)	-	[3]1 Cycle	10~32 Cycles
Annealing Time (ms)	0.05 (6.4K), [1]1 (144K)	10	-	4
Spin Area (μm^2/spin)	48 (1b-J), 216 (2b-J)	289	832	552
Power/Spin (μW/spin)	0.11	2.83	0.18	-
Energy/Spin (fJ/spin)	13.3 (6.4K), [1]6 (144K) [1]16.6 (2b-J, 6.4K) (V_{DD}=1V)	28.3	2.85 (V_{DD}=0.6V), 11.58 (V_{DD}=1.2V)	-

[1]Estimation for spins with increased parallel computations and larger memory array. [2]Simulation demonstration. [3]1 cycle for only spin update (not including computation cycles)

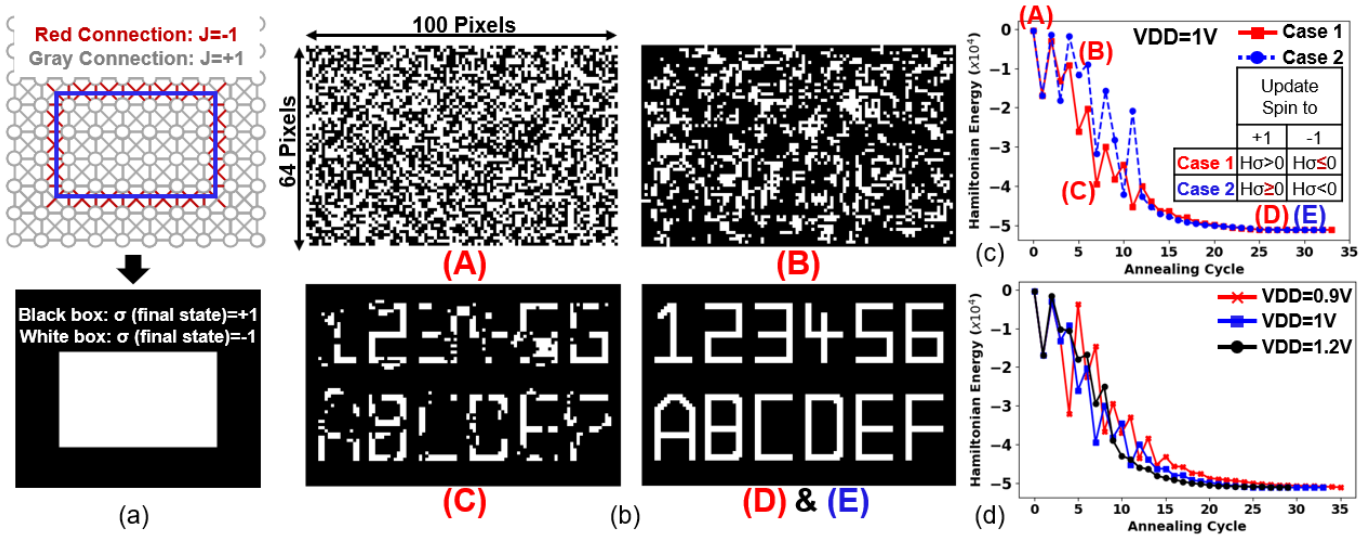


Fig. 19. (a) Spins and coefficients mapping. (b) Hamiltonian energy evolution with spin map transitions (A∼D/E) for a max-cut problem ("123456ABCDEF" image) with two spin update uncertainty cases (c) showing two annealing cycles difference for solving the same COP and (d) with three different supply voltages (0.9, 1, and 1.2 V).

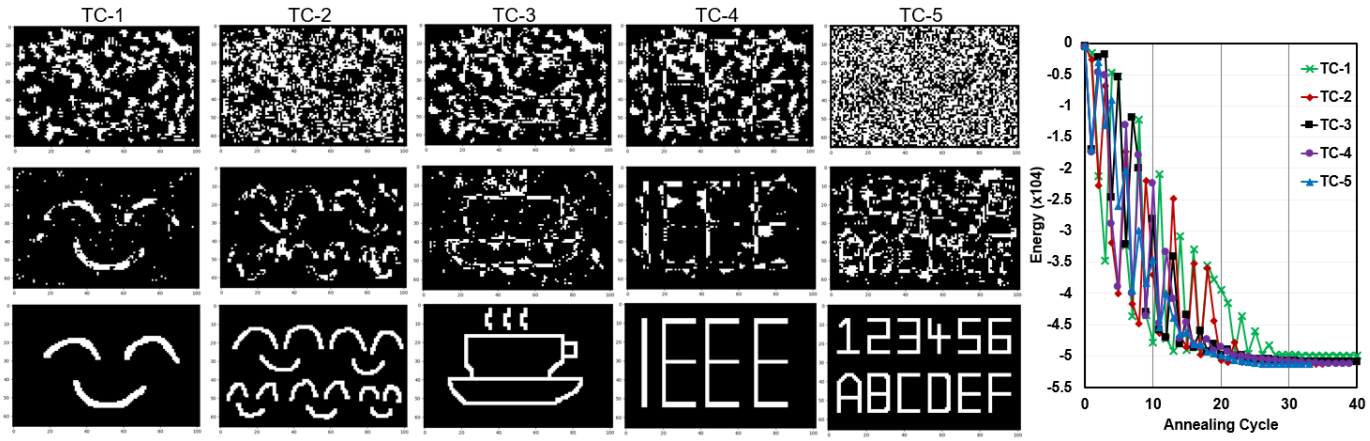


Fig. 20. Hamiltonian energy evolution for five 100 × 64 test cases. (TC-1∼TC-5.)

graph [7] and map to +1 when there is no edge crossing the interaction (gray connection). Fig. 19(b) and (c) shows the spin map and the energy evolution for two cases. In the spin map, the black box indicates $\sigma = +1$, and the white box indicates $\sigma = -1$. The experiment is captured by using two different dies. These two experiments have the same initial conditions (i.e., initial spin array, the same seed, and problem set), but the spin is updated differently depending on the SA comparison result, temperature scheduling, and energy in each annealing step. The spin update uncertainty cases (case 1 and case 2) follow slightly different energy paths in reaching the minimum energy state with a difference of 2 annealing cycles. In both cases, even though the annealing time is different, the given COP can be solved correctly from (A) to (D)/(E). In addition, voltage variation experiments on solving the same max-cut problem are shown in Fig. 19(d). Under three different supply voltages, the max-cut problem can be solved successfully with different annealing cycles. Additional five test cases (TC1–TC5) on the max-cut problem are shown in Fig. 20. This result indicates that the SA offset fluctuation due to process and temperature variations has a marginal impact on the annealing time to reach the final minimum energy state for the given size of COP.

C. Comparison With Prior Annealing Processors

Table IV compares the proposed Ising-CIM approach with prior multi-bit J coefficient AP designs [7], [13], [14], using a max-cut COP as a benchmark. The proposed Ising-CIM approach performs the analog Hamiltonian computation within a memory array. Compared with prior methods, the Ising-CIM approach minimizes extra arithmetic circuitry, significantly reducing the area overhead and improving array efficiency. The area occupied of each spin in the proposed Ising-CIM approach is 6×∼17× smaller than the prior methods [7], [13], [14]. In addition, the annealing time for solving the max-cut COPs within the test-chip memory capacity is 51.2 μs. To compare with prior approaches, software simulation is performed to determine the number of annealing cycles for a larger problem size. The corresponding annealing time is calculated based on the measured computational time

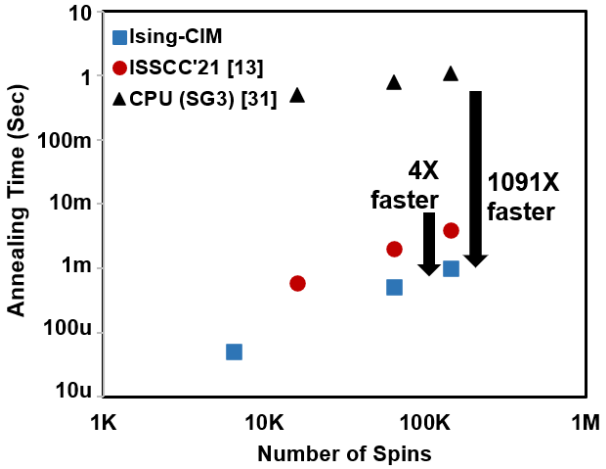


Fig. 21. Annealing time comparison between prior work [13] and CPU [31].

for each iteration and the number of annealing cycles obtained from the software simulations for a given problem. Compared to CPU annealing time [31], the speedup in Ising-CIM design is estimated to increase up to 1091× as the number of spins increases the 144 K, as shown in Fig. 21.

V. Conclusion

In this article, an Ising-CIM design with an analog CIM approach is demonstrated in a 65-nm CMOS silicon prototype. The design reduces off-chip data movement by performing most of the Ising Hamiltonian computations inside an embedded memory. Moreover, this design realizes the Hamiltonian computations without additional digital arithmetic circuits by reconfiguring the available bitcells and peripheral memory circuits. This ensures that the area overhead for embedding Ising computations within a memory array is minimized. The "ghost cell" concept is leveraged to map a large COP by splitting a large King's graph into smaller segments, mapping them onto multiple sub-arrays. Furthermore, the Ising-CIM design supports multi-bit J coefficients for more complex problem classes. Silicon prototype measurements confirm the feasibility of the proposed Ising-CIM approach and achieve a 6∼17× smaller spin-area and 4× faster annealing time for a given max-cut COP compared to prior methods. Thus, the proposed Ising-CIM design can be a promising approach to realize future energy-efficient combinatorial optimization accelerators.

Acknowledgment

The authors would like to thank S. S. Teja Nibhanupudi for helping with the test-chip micrograph setup and Dr. Andrew Lanham for the helpful discussion on annealing algorithms. They would also like to thank the TSMC University Shuttle Program for the test-chip fabrication support.

References

[1] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Chelmsford, MA, USA: Courier Corporation, 1998.

[2] K. Leppek *et al.*, "Combinatorial optimization of mRNA structure, stability, and translation for RNA-based therapeutics," *Nature Commun.*, vol. 13, no. 1, pp. 1–22, Dec. 2022. [Online]. Available: https://www.biorxiv.org/content/early/2021/03/30/2021.03.29.437587

[3] N. Pardi, M. J. Hogan, F. W. Porter, and D. Weissman, "mRNA vaccines-a new era in vaccinology," *Nature Rev. Drug Discovery*, vol. 17, no. 4, pp. 261–279, 2018.

[4] Y. Crama, "Combinatorial optimization models for production scheduling in automated manufacturing systems," *Eur. J. Oper. Res.*, vol. 99, no. 1, pp. 136–153, May 1997.

[5] K. G. Kempf, "Control-oriented approaches to supply chain management in semiconductor manufacturing," in *Proc. Amer. Control Conf.*, 2004, pp. 4563–4576.

[6] K. Benidis, Y. Feng, and D. P. Palomar, "Optimization methods for financial index tracking: From theory to practice," *Found. Trends Optim.*, vol. 3, no. 3, pp. 171–279, 2018.

[7] Y. Su, H. Kim, and B. Kim, "CIM-spin: A scalable CMOS annealing processor with digital in-memory spin operators and register spins for combinatorial optimization problems," *IEEE J. Solid-State Circuits*, early access, Jan. 17, 2022, doi: 10.1109/JSSC.2021.3139901.

[8] A. Lucas, "Ising formulations of many NP problems," *Frontiers Phys.*, vol. 2, p. 5, Feb. 2014, doi: 10.3389/FPHY.2014.00005.

[9] R. Peierls, "On Ising's model of ferromagnetism," in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 32, no. 3. Cambridge, U.K.: Cambridge Univ. Press, 1936, pp. 477–481.

[10] C. Yoshimura *et al.*, "Uncertain behaviours of integrated circuits improve computational performance," *Sci. Rep.*, vol. 5, no. 1, pp. 1–12, Dec. 2015.

[11] T. Takemoto, M. Hayashi, C. Yoshimura, and M. Yamaoka, "2.6 A 2 ×30k-spin multichip scalable annealing processor based on a processing-in-memory approach for solving large-scale combinatorial optimization problems," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2019, pp. 52–54.

[12] K. Yamamoto *et al.*, "STATICA: A 512-spin 0.25M-weight annealing processor with an all-spin-updates-at-once architecture for combinatorial optimization with complete spin-spin interactions," *IEEE J. Solid-State Circuits*, vol. 56, no. 1, pp. 165–178, Jan. 2021.

[13] T. Takemoto *et al.*, "4.6 A 144Kb annealing system composed of 9×16Kb annealing processor chips with scalable chip-to-chip connections for large-scale combinatorial optimization problems," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 64–66.

[14] M. Yamaoka, C. Yoshimura, M. Hayashi, T. Okuyama, H. Aoki, and H. Mizuno, "A 20k-spin ising chip to solve combinatorial optimization problems with CMOS annealing," *IEEE J. Solid-State Circuits*, vol. 51, no. 1, pp. 303–309, Jan. 2016.

[15] S. Xie, C. Ni, A. Sayal, P. Jain, F. Hamzaoglu, and J. P. Kulkarni, "16.2 eDRAM-CIM: Compute-in-memory design with reconfigurable embedded-dynamic-memory array realizing adaptive data converters and charge-domain computing," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 248–249.

[16] F. B. Kjolstad and M. Snir, "Ghost cell pattern," in *Proc. Workshop Parallel Program. Patterns (ParaPLoP)*, 2010, pp. 1–9.

[17] S. G. Brush, "History of the Lenz–Ising model," *Rev. Modern Phys.*, vol. 39, no. 4, p. 883, 1967.

[18] M. W. Johnson *et al.*, "Quantum annealing with manufactured spins," *Nature*, vol. 473, no. 7346, pp. 194–198, May 2011.

[19] D. Pierangeli, G. Marcucci, and C. Conti, "Large-scale photonic Ising machine by spatial light modulation," *Phys. Rev. Lett.*, vol. 122, no. 21, May 2019, Art. no. 213902, doi: 10.1103/PhysRevLett.122.213902.

[20] T. Wang, L. Wu, and J. Roychowdhury, "New computational results and hardware prototypes for oscillator-based Ising machines," in *Proc. 56th Annu. Design Autom. Conf.*, Jun. 2019, pp. 1–2.

[21] *King's Graph*. Accessed: Nov. 21, 2021. [Online]. Available: https://en.wikipedia.org/wiki/King's_graph

[22] C. Cook, H. Zhao, T. Sato, M. Hiromoto, and S. X.-D. Tan, "GPU-based Ising computing for solving max-cut combinatorial optimization problems," *Integration*, vol. 69, pp. 335–344, Nov. 2019.

[23] T. Wang and J. Roychowdhury, "PHLOGON: Phase-based logic using oscillatory nano-systems," in *Proc. Int. Conf. Unconventional Comput. Natural Comput.*, Cham, Switzerland: Springer, 2014, pp. 353–366.

[24] G. Csaba and W. Porod, "Noise immunity of oscillatory computing devices," *IEEE J. Explor. Solid-State Comput. Devices Circuits*, vol. 6, pp. 164–169, 2020.

[25] I. Ahmed, P.-W. Chiu, and C. H. Kim, "A probabilistic self-annealing compute fabric based on 560 hexagonally coupled ring oscillators for solving combinatorial optimization problems," in *Proc. IEEE Symp. VLSI Circuits*, Jun. 2020, pp. 1–2.
[26] D. Landau and K. Binder, *A Guide to Monte Carlo Simulations in Statistical Physics*. Cambridge, U.K.: Cambridge Univ. Press, 2021.
[27] R. W. Mann, "Interactions of technology and design in nanoscale SRAM," Ph.D. dissertation, Dept. School Eng. Appl. Sci., Univ. Virginia, Charlottesville, VA, USA, 2010.
[28] G. Van Rossum and F. L. Drake, Jr., *Python Tutorial*, vol. 620. Amsterdam, The Netherlands: Centrum voor Wiskunde en Informatica Amsterdam, 1995.
[29] *Xilinx Virtex-7 FPGA VC707 Evaluation Kit*. Accessed: Jun. 1, 2021. [Online]. Available: https://www.xilinx.com/products/boards-and-kits/ek-v7-vc707-g.html
[30] *FMC XM105 Debug Card*. Accessed: Jun. 1, 2021. [Online]. Available: https://www.xilinx.com/products/boards-and-kits/hw-fmc-xm105-g.html
[31] S. Kahruman, E. Kolotoglu, S. Butenko, and I. V. Hicks, "On greedy construction heuristics for the MAX-CUT problem," *Int. J. Comput. Sci. Eng.*, vol. 3, no. 3, pp. 211–218, 2007.

Shanshan Xie (Graduate Student Member, IEEE) received the B.S degree in electrical and computer engineering (ECE) from the Worcester Polytechnic Institute (WPI), Worcester, MA, USA, in 2018. She is currently pursuing the M.S. and Ph.D. degrees in electrical and computer engineering with The University of Texas at Austin (UT Austin), Austin, TX, USA.

She was an Intern with Analog Devices, Wilmington, MA, USA, and Texas Instrument Corporation, Dallas, TX, USA, where she was involved in programmable gain instrumentation amplifier, electrocardiogram (ECG) heart rate monitor, and controller area network (CAN) isolation products. Her research interests include the mixed-signal design for compute-in-memory techniques, machine learning accelerators, and annealing processors.

Ms. Xie was a recipient of the Cadence Women in Technology Scholarship from Cadence in 2020. She was one of the 2021-2022 SSCS Predoctoral Achievement Award Recipients.

Siddhartha Raman Sundara Raman (Member, IEEE) received the B.S. degree from the Birla Institute of Technology and Science, Pilani, India, in 2019, and the M.S. degree from The University of Texas at Austin, Austin, TX, USA, in 2021.

He interned at the National University of Singapore, Singapore, on spintronic circuits, and NVIDIA, Bengaluru, India, as part of his undergraduate study. His current research interests include compute-in-memory, cryogenic computing, and device-to-circuit optimization for emerging non-volatile memory (NVM) technologies.

Can Ni (Member, IEEE) received the bachelor's degree in engineering physics from the University of Alberta, Edmonton, AB, Canada, in 2019, and the master's degree in electrical and computer engineering from The University of Texas at Austin, Austin, TX, USA, in 2021.

He was an Intern with Qualcomm, San Diego, CA, USA. His research interests include compute-in-memory and neuromorphic hardware.

Meizhi Wang (Graduate Student Member, IEEE) received the B.S. degree in electrical engineering, the B.S. degree in system science and engineering, and the M.S. degree in electrical engineering from the Washington University in St. Louis, St. Louis, MO, USA, in 2018. She is currently pursuing the Ph.D. degree with the Electrical and Computer Engineering Department, The University of Texas at Austin, Austin, TX, USA, specializing in the integrated circuits and systems track.

Her research interests include hardware security and low-power VLSI design.

Mengtian Yang received the B.S. degree in computer science from Shanghai Jiao Tong University, Shanghai, China, in 2021. He is currently pursuing the Ph.D. degree in electrical and computer engineering with The University of Texas at Austin, Austin, TX, USA, specializing in the architecture, computer systems, and embedded systems track.

He is currently with the Circuits Research Laboratory, The University of Texas at Austin. His research interests include hardware acceleration for combinational optimization problems, computer-in-memory, and machine learning accelerators.

Jaydeep P. Kulkarni (Senior Member, IEEE) received the B.E. degree from the University of Pune, Pune, India, in 2002, the M.Tech. degree from the Indian Institute of Science (IISc), Bengaluru, India, in 2004, and the Ph.D. degree from Purdue University, West Lafayette, IN, USA, in 2009, all in electronics/electrical engineering.

From 2009 to 2017, he was with the Intel Circuit Research Laboratory, Hillsboro, OR, USA, where he worked on energy-efficient integrated circuit technologies. He is currently an Assistant Professor of electrical and computer engineering with The University of Texas at Austin, Austin, TX, USA, where he is also a fellow of the AMD Endowed Chair in Computer Engineering and the Silicon Labs Chair in Electrical Engineering. He has filed 35 patents and published 100 articles in referred journals and conferences. His research is focused on machine learning hardware accelerators, in-memory computing, emerging nano-devices, hardware security, heterogeneous/3-D integration, and cryogenic computing.

Dr. Kulkarni is a member of the Association for Computing Machinery (ACM). He received the Best M.Tech. Student Award from IISc, the Intel Foundation Ph.D. Fellowship Award, the SRC Best Paper and Inventor Recognition Awards, the Purdue Outstanding Doctoral Dissertation Award, seven Intel Divisional Recognition Awards, the 2015 IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS Best Paper Award, the SRC Outstanding Industrial Liaison Award, the Micron Foundation Faculty Awards, the Intel Rising Star Faculty Award, and the NSF Career Award. He has served as the Conference General Co-Chair of 2018 International Symposium on Low Power Electronics and Design (ISLPED) and is participating in the technical program committees of the Custom Integrated Circuits Conference (CICC), International Conference on Computer-Aided Design (ICCAD), Design Automation Conference (DAC), and International Conference on Artificial Intelligence Circuits and Systems (AICAS) conferences. He is also serving as the Chair of the IEEE Central Texas SSCS/CAS Joint Chapter. He is also serving as an Associate Editor for the IEEE SOLID-STATE CIRCUITS LETTERS, the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, and IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS.