RRAM-Based Computing-In-Memory FPGA Architecture for Low-Power Applications

Xiaochang Zhang, Yongqiang Zhong, Daoda An, Yewei Zhang, Pengpeng Ren, *Member, IEEE*, Runsheng Wang, *Senior Member, IEEE*, Yimao Cai, *Member, IEEE*, Feng Hao, *Senior Member, IEEE*, and Zhigang Ji, *Member, IEEE*

***Abstract*—Circuit design often faces a fundamental trade-off: ASICs offer peak efficiency at the cost of high design complexity and inflexibility, whereas FPGAs provide fast adaptability but suffer from high power consumption. In this work, we propose an RRAM-based FPGA architecture with a Reconfigurable Computing-in-Memory Logic Unit (RCLU), thereby achieving both flexibility, low cost, and low power consumption. By proposing a 2T2R structure that prevents RESET/SET operations in RRAM during logic computation, the proposed RCLU achieves significant improvements in delay and energy efficiency. In experimental benchmarks against conventional SRAM-FPGA and state-of-the-art RRAM-FPGA design, our architecture achieves power-delay product (PDP) improvements of up to 76.89% and 57.09%, respectively. Furthermore, to validate the low-power advantage of the proposed architecture, we implement ASCON, a representative lightweight IoT encryption algorithm. Simulation results show that our design achieves significantly lower energy per bit than conventional FPGAs and approaches ASIC-level efficiency, demonstrating strong potential as an energy-efficient IoT hardware solution.**

***Index Terms*—RRAM, FPGA, Computing-In-Memory, Internet of Things.**

I. INTRODUCTION

With the widespread deployment of Internet of Things (IoT) devices in areas such as smart homes, industrial monitoring, and healthcare, the demand for high energy efficiency and flexibility in edge devices has become increasingly prominent [1][2]. IoT edge devices are often deployed in resource-constrained environments (e.g., battery-powered sensor nodes). For example, processing complex tasks on edge devices often entails intensive computation, thereby significantly increasing power consumption and shortening device lifetimes [3]. Moreover, as IoT application scenarios continue to evolve rapidly, edge devices must not only meet stringent low-power requirements but also remain adaptable to diverse and evolving workload. Consequently, designing hardware architectures that combine low energy consumption with reconfigurability has emerged as a critical research direction, aiming to provide more efficient solutions for edge computing in IoT [4]-[6].

Current hardware solutions for IoT edge devices primarily include Application-Specific Integrated Circuits (ASICs) and Field-Programmable Gate Arrays (FPGAs). ASICs, which are custom designed for specific tasks, typically offer advantages over FPGAs in performance, power, and area (PPA) because they can be deeply optimized. However, ASIC development is complex, with high design, verification, and test costs, accompanied by long development cycles. Once manufactured, the architecture of an ASIC becomes fixed. Therefore, conventional ASIC-based solutions struggle to adapt to the evolving demands of IoT applications through updates.

FPGAs represent another widely adopted solution for IoT edge devices. As programmable semiconductor devices, they enable rapid realization of customized hardware designs with high flexibility and reconfigurability. However, traditional SRAM-based FPGAs suffer from high power consumption, large area overhead, and volatility (i.e., data loss upon power-off) [4]. These limitations are largely attributable to the inherent characteristics of SRAM. The advent of emerging non-volatile memory technologies offers potential to overcome these limitations. Representative candidates include Spin-Transfer Torque Magnetic RAM (STT-MRAM) [7], Phase Change Memory (PCM) [8], and Resistive Random-Access Memory (RRAM) [9]. Among these, RRAM has garnered significant attention due to its CMOS compatibility, low power characteristic, compact size, and high density. These advantages position RRAM as one of the most promising memory solutions for IoT edge devices [10][11].

To address both low-power and reconfigurability requirements, RRAM-based FPGA architectures have been proposed. Current approaches generally fall into the following categories: (1) using RRAM to replace both the SRAM in LUTs of traditional FPGAs and the configuration SRAM that controls channel switches in programmable interconnects, leveraging RRAM's memory characteristics. [12]; (2) using RRAM as programmable switches in place of SRAM-controlled MUXs, leveraging the resistance switching characteristics of RRAM (i.e., HRS for OFF, LRS for ON) [13]; and (3) hybrid architectures combining both SRAM and RRAM to exploit their respective advantages [14]. These research efforts will be further discussed in Section II. However, it is worth noting that existing works primarily focus on replacing specific components within conventional FPGA architectures with RRAM-based alternatives, while the

This work was supported in part by the National Natural Science Foundation of China under Grant No. 92164205.

Xiaochang Zhang, Yongqiang Zhong, Daoda An, Yewei Zhang, Pengpeng Ren and Zhigang Ji are with the National Key Laboratory of Science and Technology on Micro/Nano Fabrication, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: zhigangji@sjtu.edu.cn).

Runsheng Wang and Yimao Cai are with the Institute of Microelectronics, Peking University, Beijing 100871, China.

Feng Hao is with the Department of Computer Science, University of Warwick, Coventry CV4 7AL, U.K.

underlying logic computation principles and the data movement between memory arrays and logic control circuits remain unchanged. As such, the energy overhead caused by frequent data transfer persists.

Building upon these works, this paper proposes a further step toward reducing energy consumption by introducing the concept of computing-in-memory (CIM). Unlike conventional FPGAs, we propose a novel RRAM-based Computing-In-Memory Logic Unit (RCLU) that leverages the advantages of CIM. Compared to traditional FPGAs, the proposed RCLU-based architecture offers significant improvements in energy and area efficiency. The main contributions of this paper are as follows:

We propose an RCLU-based RRAM-FPGA architecture for low-power reconfigurable computing. Different from conventional SRAM-LUTs and storage-replacement RRAM-LUTs, the proposed RCLU does not store and select a complete truth table. Instead, the pre-initialized RRAM states directly participate in logic evaluation through resistive voltage division, forming a localized logic-in-memory primitive at the FPGA logic-block level.

We redesign FELIX [15]-inspired stateful RRAM logic into an FPGA-compatible voltage-sensing logic unit. By introducing a 2T2R selection structure, the RCLU converts input-dependent RRAM switching into voltage-domain branch selection and avoids SET/RESET operations during normal computation. As a result, the output is determined by the internal sensing voltage V_Y and converted to a full-swing digital signal by CMOS buffering, which reduces switching latency and improves compatibility with downstream FPGA routing and sequential circuits.

We evaluate the robustness and limitations of the proposed RCLU under representative device and circuit non-idealities. The analysis considers RRAM device-to-device variation, LRS conductance drift, transistor process variation, operating-voltage fluctuation, temperature variation, aging-induced transistor degradation, and cycle-to-cycle variation. The results show that the two critical input cases still satisfy the output-buffer noise-margin requirement under the evaluated conditions.

We implement and evaluate the proposed architecture using a complete FPGA CAD flow. The proposed RRAM-FPGA is benchmarked against conventional SRAM-FPGA and prior RRAM-FPGA designs in terms of power-delay product, programming energy, area, and routing overhead. In addition, ASCON, a representative lightweight IoT encryption algorithm, is mapped to the proposed architecture to demonstrate its potential for energy-efficient and reconfigurable IoT edge hardware.

The remainder of this paper is organized as follows: Section II introduces RRAM and conventional FPGA architectures, reviews existing RRAM-based FPGA works, and highlights their limitations. Section III presents the design of the proposed RRAM-based FPGA architecture with the RCLU. Section IV provides experimental setup and benchmarking results. Section V explores the application of the proposed architecture to lightweight IoT encryption algorithms. Section VI concludes the paper.

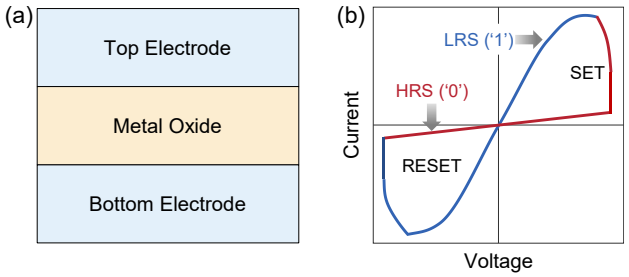


Fig. 1. (a) RRAM structure; (b) Bipolar RRAM I-V characteristics [17].

TABLE I
PERFORMANCE COMPARISON OF NVM [18]

	Cell area	Read time	Write time	Retention	Endurance	Write energy(J/bit)
STT-MRAM	6~50 F^2	<10ns	<10ns	>10y	>1E15	~0.1pJ
PCRAM	4~30 F^2	<10ns	~50ns	>10y	>1E9	~10pJ
RRAM	4~12 F^2	<10ns	<10ns	>10y	>1E6~1E12	~0.1pJ

II. BACKGROUND AND MOTIVATIONS

A. Resistive Random-Access Memory (RRAM)

Resistive Random-Access Memory is an emerging non-volatile memory (NVM) technology, typically composed of a metal–insulator–metal (MIM) structure, as illustrated in Fig. 1(a). By applying electrical stimuli across the terminals of the device, its resistance state can be switched between a high-resistance state (HRS) and a low-resistance state (LRS), thereby enabling data storage. Conventionally, HRS and LRS are used to represent the binary values "0" and "1," respectively. Due to its simple structure and ease of integration, RRAM has attracted significant attention in both academia and industry.

A widely used type of RRAM is the conductive filament (CF)-based variant, where resistive switching is achieved via the formation and rupture of conductive filaments composed of oxygen vacancies between two electrodes. This paper adopts an oxide-based CF-RRAM device in which the conductive paths are formed through the aggregation of oxygen vacancies.

Fig. 1(b) shows the typical I–V characteristics of a bipolar RRAM cell. The switching from LRS to HRS is referred to as the "RESET" process (i.e., writing a logic "0"), while the transition from HRS to LRS is called "SET" (i.e., writing a logic "1"). In bipolar devices, the switching behavior depends on the polarity of the applied voltage. During RESET, oxygen ions migrate into the oxide layer and recombine with the oxygen vacancies, rupturing the conductive filament. During SET, the oxygen vacancies are re-separated and re-aggregated

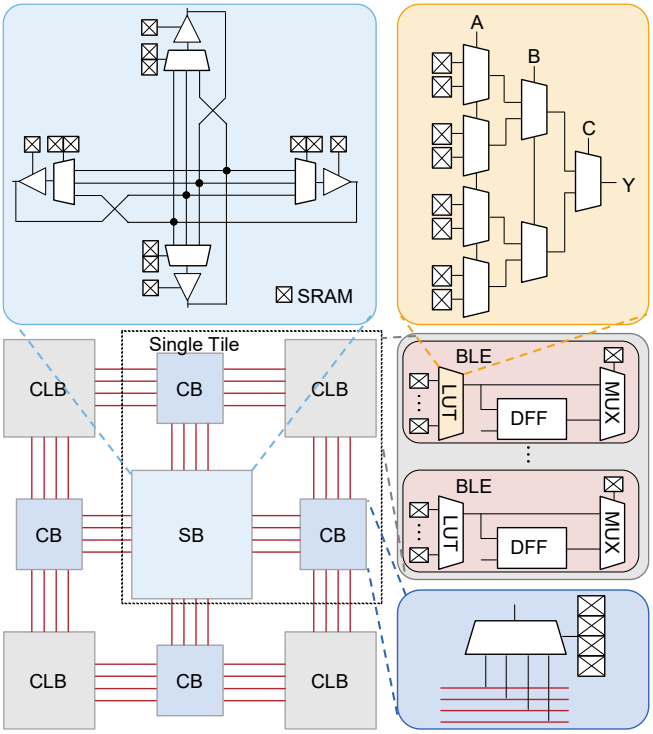


Fig. 2. Traditional SRAM-Based FPGA architecture, including CLBs, SBs, and CBs.

to reform the filament [17].

An important advantage of RRAM is that it can be fabricated using Back-End-of-Line (BEoL) processes between metal layers, enabling extremely high memory density. Table I presents a performance comparison of three types of NVMs. Given its fast read/write speed (<10 ns), compact cell size (4~12 F^2), low programming energy (~0.1 pJ), long retention (>10 years), and compatibility with CMOS processes, RRAM is a highly suitable candidate for IoT edge device [18].

B. Conventional FPGA Architecture

Traditional FPGA architectures are commonly based on island-style structures [13], as illustrated in Fig. 2. A typical FPGA consists of replicated tiles, each comprising a configurable logic block (CLB), a switch block (SB), and connection blocks (CBs). The SB and CB together form the programmable interconnect (PI). Each CLB contains one or more basic logic elements (BLEs), which include look-up tables (LUTs), flip-flops, and multiplexers. As shown in Fig. 2, a K-input LUT stores all possible output combinations of a Boolean function using 2^K bits of configuration memory, typically implemented with SRAM cells. The inputs to the LUT control a multiplexer network that selects the corresponding stored value as the output. After the LUT, flip-flops and local interconnects (MUXs) provide connectivity between BLEs.

Unlike ASICs where interconnects are fixed, FPGAs use programmable switches for connectivity. While this enables reconfigurability, it also results in higher area and power consumption, making traditional FPGAs less favorable for low-power applications such as portable or IoT edge devices. The interface between CLBs and the routing channels is managed by CBs, while SBs control routing at channel intersections using programmable switches. These switches are implemented with MUXs controlled by SRAM cells, similar in concept to the LUTs.

However, SRAM-based FPGAs face several critical limitations. SRAM cells are volatile, consume significant leakage power (especially at advanced technology nodes), and suffer from low density. The most advanced SRAM-based FPGAs today contain over 10^8 SRAM bits, with static leakage accounting for up to 60% of total power consumption. Excessive buffers are often required to ensure signal integrity, resulting in large area, power, and delay overheads for the programmable interconnect network [19].

C. RRAM-Based FPGA Architectures

RRAM-based FPGAs have been proposed to overcome the limitations of traditional SRAM-based FPGAs in terms of power, area, and volatility. Existing works mainly exploit the non-volatility, BEOL compatibility, and high integration density of RRAM to replace configuration memories or programmable routing resources in FPGA fabrics.

One research direction focuses on replacing the SRAM cells in LUTs and interconnect switches with RRAM devices [12]. Due to its 3D stackability and high density, RRAM can significantly reduce the configuration-memory area. Its non-volatility also eliminates the need to reload configuration data on power-up, saving static and configuration energy. For example, Gaillardon et al. [20] proposed a 3D-stacked RRAM-based FPGA that achieved a 28% area reduction and a 34% delay reduction compared to conventional FPGAs. Tanachutiwat et al. proposed a CMOS-RRAM hybrid reconfigurable FPGA architecture based on 1T1R RRAM structures, where RRAM devices are used for FPGA block memories.[9] Chen et al. proposed a non-volatile 3D-stacked RRAM-based FPGA, where 1R memory cells are used in LUTs and complementary resistive switches are used in routing elements [39] These works mainly use RRAM as dense and nonvolatile configuration storage while preserving the conventional LUT-based logic-evaluation mechanism.

Another approach uses RRAM devices as resistive programmable routing switches along the data transmission paths. In this case, HRS corresponds to an OFF state, and LRS corresponds to ON. Since RRAM retains its state after power loss, the configuration is preserved without the need for reload. Cong et al. demonstrated a transistor-less programmable interconnect structure using RRAM switches, achieving a 96% area reduction, 55% performance improvement, and 79% power reduction over traditional architectures [13]. Tang et al. investigated RRAM-based multiplexers and programmable interconnects for low-power FPGA fabrics, including near-threshold operation, one-level RRAM-based MUX structures, and different RRAM programming schemes [12][29][36]. These studies show that RRAM-based routing elements can effectively reduce routing

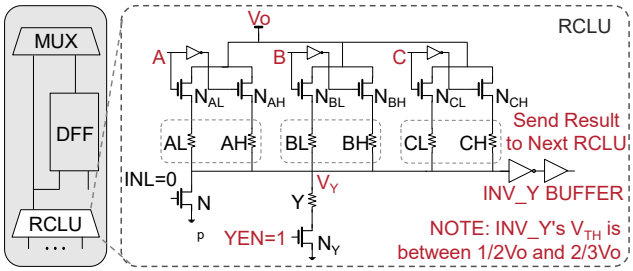


Fig. 3. Circuit structure and computing mechanism of RCLU.

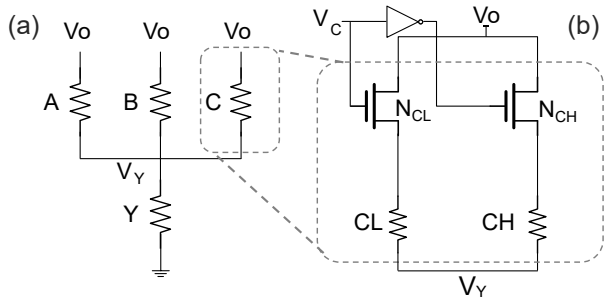


Fig. 4. (a) FELIX logic; (b) Proposed 2T2R structure.

overhead, but the logic block itself still follows the conventional LUT or MUX-based computation principle.

A third approach combines SRAM and RRAM in hybrid architectures, placing SRAM along critical paths to minimize delay, and using RRAM on non-critical paths to reduce leakage power [14]. The challenge lies in developing effective placement algorithms that optimize the SRAM/NVM trade-off based on path criticality.

Several studies have also evaluated RRAM-FPGA architectures from the full-fabric, programming, and reliability perspectives. Zambelli et al. analyzed the trade-off between programming strategy, power efficiency, and lifetime in RRAM-based FPGAs, indicating that aggressive programming can improve the HRS/LRS window but may reduce device lifetime [37]. Rizzi et al. investigated process-voltage-temperature variation and time-dependent conductance drift in RRAM-based FPGA MUX designs, highlighting the importance of considering RRAM non-idealities in practical FPGA deployment [38].

The aforementioned RRAM-based FPGA architectures can be uniformly categorized as storage-replacement designs: RRAM devices are adopted either as nonvolatile configuration memory or as resistive routing switches, while core logic evaluation is still performed by CMOS multiplexer networks. In such designs, RRAM only contributes to nonvolatility and density improvement, without directly participating in logic computation. The underlying computing paradigm remains identical to conventional SRAM-FPGAs, and the energy overhead caused by data movement between storage and logic circuits is not fundamentally addressed.

Beyond storage-replacement RRAM-FPGA designs, stateful RRAM logic has demonstrated that RRAM resistance states can directly participate in Boolean-function evaluation, providing a useful reference for exploring logic mechanisms beyond conventional LUT-based computation. Representative stateful logic structures such as FELIX [15] demonstrate the feasibility of implementing Boolean operations within resistive memory arrays. However, such designs rely on repeated RRAM SET/RESET operations during each logic evaluation, leading to high latency and poor robustness against device variations, which makes them unsuitable for direct use as FPGA logic primitives.

This work proposes a RCLU as the basic logic element for RRAM-based FPGAs. Different from storage-replacement RRAM-LUTs, the RRAM devices in the proposed RCLU form a resistive computing network and directly participate in logic evaluation via voltage division, representing a localized form of CIM at the FPGA logic-block level. Compared with stateful CIM logic such as FELIX, the proposed RCLU eliminates runtime RRAM switching and provides voltage-domain outputs compatible with standard FPGA routing and sequential circuits.

III. CIM EMBEDDED RRAM-FPGA ARCHITECTURE

A. RRAM-based Computing-In-Memory Logic Unit (RCLU)

Similar to conventional SRAM-based FPGAs, our proposed architecture adopts an island-style structure comprising logic blocks, switch blocks, and connection blocks.

The primary innovation of our work lies in the logic block. We propose a novel RCLU (Fig. 3) to replace the traditional SRAM-based LUT. This structure combines the benefits of CIM and lookup-table-based logic, providing significant improvements in both power and area. Section IV will present the detailed experimental evaluation. This section focuses on the design of RCLU, including its computing mechanism, initialization principle, and a brief analysis of its reliability.

Before formally introducing the structure of RCLU, we first provide a brief overview of the FELIX [15] CIM logic computing principle, proposed by Saransh Gupta et al., upon which RCLU builds with innovations. The three-input FELIX logic structure is shown in Fig. 4(a), consisting of four RRAMs, with resistance states used as logic variables—HRS defined as logic "0" and LRS as logic "1." A, B, and C are input units, while Y is the output unit. When Vo is applied to one end of A, B, and C, V_Y represents the voltage division at Y.

FELIX logic employs the principle of resistive voltage division. Before computation, the RRAMs (A, B, C) are preset with resistance states corresponding to the input logic, while the output Y is preset to LRS. During operation, one end of the three RRAMs is connected to voltage Vo, and the output unit is grounded. The node voltage V_Y is influenced by voltage division, with different input states corresponding to different voltages.

When all three input units are HRS (input combination HHH), the V_Y node voltage is close to GND. When only one of the three inputs is LRS (input combinations equivalent to

TABLE II
TRUTH TABLE OF FELIX LOGIC

Input ABC	V_Y	Output Y
HHH	GND	L
LHH	1/2Vo	L
LLH	2/3Vo	H
LLL	3/4Vo	H

TABLE III
ENABLING THREE LOGIC OPERATIONS VIA PRE-PROGRAMMED C

Input ABC	Logic Operation
XXL	$\overline{A+B}$ (NOR)
XXH	$\overline{AB}$ (NAND)
XLH	$\overline{A}$ (NOT)

LHH), since HRS ≫ LRS, the HRS in the parallel structure acts as an open circuit, and the circuit is equivalent to two LRS resistors in series, resulting in a V_Y node voltage of approximately 1/2Vo. When two of the three inputs are LRS (input combinations equivalent to LLH), the circuit is equivalent to two LRS resistors in parallel connected in series with one LRS resistor, yielding a V_Y node voltage of 2/3Vo. When all three input units are LRS (input combination LLL), the circuit is equivalent to three LRS resistors in parallel connected in series with one LRS resistor, resulting in a V_Y node voltage of 3/4Vo. In the original design, when the inputs are HHH and LHH, the output Y remains LRS; when the inputs are LLH and LLL, the output Y is RESET to HRS. The logical expression for the three-input FELIX is $Y = \overline{AB + BC + AC}$, and its truth table is shown in Table II.

Additionally, as shown in Table III, by preset input C to LRS (logic "1"), the three-input FELIX can implement a two-input NOR: $Y = \overline{A+B}$. By preset input C to HRS (logic "0"), the three-input FELIX can implement a two-input NAND: $Y = \overline{AB}$. By preset input B to LRS (logic "1") and C to HRS (logic "0"), the three-input FELIX can implement NOT: $Y = \overline{A}$.

FELIX is a representative stateful logic structure that demonstrates the feasibility of in-memory Boolean computation. However, two fundamental limitations prevent it from being directly adopted as a logic primitive in FPGA fabrics. First, the computation latency of FELIX is inherently high and variable. In FELIX logic, the output is encoded as the resistance state of the output RRAM device, which requires an intentional SET or RESET switching operation during each logic evaluation. RRAM switching typically takes several nanoseconds and exhibits large cycle-to-cycle variability, which is incompatible with the strict timing requirements of FPGA logic paths. Second, FELIX suffers from poor robustness against device variations. Correct logic operation relies on precise control of the applied voltage to trigger switching only for specific input combinations. Practical RRAM devices exhibit significant variations in switching threshold and resistance states, which can easily lead to functional failures due to the narrow valid operation window [35]. These two issues motivate the design of the proposed RCLU, which transforms the stateful logic principle into a voltage-sensing-based logic primitive suitable for FPGA integration.

TABLE IV
TRUTH TABLE OF RCLU

ABC	V_Y	BUFFER_OUT
000	GND	1
100	1/2Vo	1
110	2/3Vo	0
111	3/4Vo	0

We designed a 2T2R (2 Transistors and 2 RRAMs) circuit structure as shown in Fig. 4(b). The function of this circuit is to convert both the inputs and outputs during computation from resistance states to voltage signals (C → V_C, Y → V_Y). This approach avoids resistance state switching in the RRAMs, thereby significantly reducing the computation delay. The details are as follows:

In the original FELIX design, if input C is logic "0" (or "1"), RRAM C must be RESET (or SET) to HRS (or LRS) during computation. To reduce latency, we replace the “RESET/SET” process with a "selection" mechanism. As shown in Fig. 4(b), the dashed box contains two preset RRAM units—CL (LRS) and CH (HRS). Depending on whether input C is logic "0" or "1," transistor N_{CH} or N_{CL} is turned on, selecting the preconfigured CH (HRS) or CL (LRS) unit. This significantly reduces computation delay. While this improvement increases area overhead compared to the original FELIX, the area remains advantageous over traditional LUTs (detailed in Section IV). In the new 2T2R circuit structure, the Vo voltage needs to be as low as possible, since a high Vo voltage may unintentionally RESET the output RRAM Y during computation (e.g., for LLH or LLL inputs). So, we reduce Vo to ensure computation relies purely on resistive voltage division—without altering RRAM states. The new logic interpretation is as follows:

Logic "1": Input combinations HHH (V_Y ≈ GND) and LHH (V_Y ≈ 1/2Vo). Logic "0": Input combinations LLH (V_Y ≈ 2/3Vo) and LLL (V_Y ≈ 3/4Vo).

The selection of Vo follows a different design objective from that in FELIX. In the original FELIX logic, Vo must be selected to intentionally trigger the output RRAM switching for the input cases that should change the output resistance state, while avoiding switching for the other input cases. In the

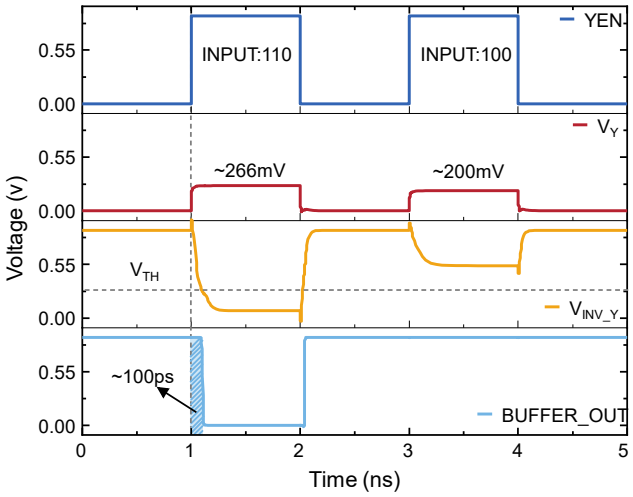


Fig. 5. Functional verification of the proposed RCLU.

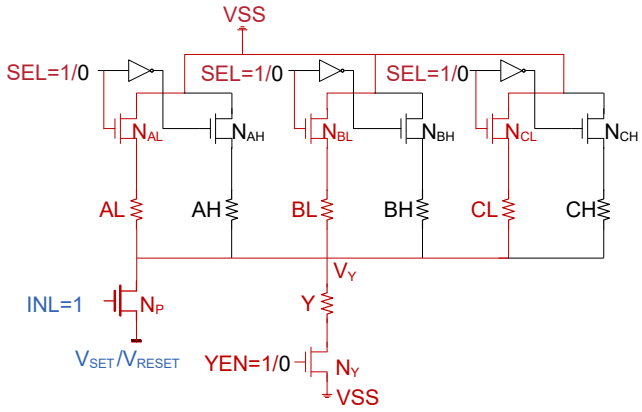


Fig. 6. Initialization mechanism of the RCLU.

proposed RCLU, the output RRAM should not be switched during normal computation. Therefore, Vo is selected under two constraints. On the one hand, Vo should be sufficiently low to prevent unintended RESET/SET disturbance of the RRAM devices. On the other hand, Vo should not be too low, because the voltage separation between the critical VY levels, especially the LHH case near 1/2Vo and the LLH case near 2/3Vo, must remain large enough for the output inverter to distinguish the logic states. Therefore, Vo is not simply minimized, but is chosen to balance RRAM non-disturbance and output sensing margin.

Integrating these optimizations, we present RCLU (Fig. 3), derived from the three-input FELIX. At fabrication, AH/BH/CH are preset to HRS, while AL/BL/CL/Y are preset to LRS. With an appropriately chosen Vo, no RRAM RESET/SET occurs during operation. Table IV presents the truth table for RCLU. Taking the NAND operation of inputs A and B as an example, let C be set to logic "0" (via connecting C to ground through an RRAM based programmable switch described in Section III-C), which activates transistor N_{CH} and selects the CH RRAM cell. Under this configuration, the circuit implements a two-input NAND logic operation. The threshold voltage V_{TH} of the inverter INV_Y is set between 1/2Vo and 2/3Vo. The simulated waveform of the circuit is shown in Fig. 5. The BUFFER is used to rapidly pull the output of INV_Y to logic "1" or "0". In the simulation, Vo is set to 400 mV (which can be slightly increased to compensate for voltage drop across the transistors), with RRAM HRS and LRS values set to 100 kΩ and 10 kΩ, respectively. When both A and B are logic "1", $V_Y \approx 2/3Vo \approx 266mV$; when either A or B is logic "1", $V_Y \approx 1/2Vo \approx 200mV$. The total delay from the start of computation to the output of the BUFFER is approximately 100 ps.

It should be noted that all simulations include the voltage drop across the NMOS selector transistors. Due to the relatively high RRAM resistance and low operating voltage, the branch current is small, and the voltage loss introduced by the on-resistance of the NMOS selectors is limited. Monte Carlo simulations show that under nominal conditions, the V_Y voltages for critical input patterns "100" and "110" are 202.8 mV and 257.9 mV, respectively, and the two voltages can still be clearly distinguished by the BUFFER.

The present RCLU is intentionally designed as a fine-grained two-input primitive. Although a higher-fan-in RCLU could support richer logic functions, it would introduce more intermediate voltage-division levels and reduce the separation between adjacent VY levels, making sensing more sensitive to RRAM variation, selector voltage drop, PVT fluctuation, and supply noise. Therefore, in order to ensure robust NAND/NOR/NOT operation, the RCLU is designed with two inputs.

B. RCLU Initialization

The configuration of RCLU is more appropriately referred to as initialization rather than programming, since all RRAMs within the RCLU are programmed only once when the FPGA is first used. During initialization, AL, BL, CL, and Y are set to the LRS, while AH, BH, and CH are reset to the HRS. Ideally, during the subsequent repeated power-on/off or operational processes of the FPGA, the RRAMs in the RCLU do not require reprogramming. This is because RCLU employs a relatively small Vo voltage during computation, and the resistance values of all RRAMs in the RCLU hardly shift during the calculation process. In contrast, programming in the conventional sense—used throughout this paper—refers to configuring the logical functions of each RCLU (any RCLU can be configured into NAND, NOR, or NOT functions according to TABLE III) and programming the programmable interconnection part.

The initialization procedure consists of only two steps (Fig. 6). During initialization, INL = 1, and the N_P transistor is turned on. When SEL = 1 and YEN = 1, all RRAMs that need to be set to LRS are selected and simultaneously initialized to LRS; when SEL = 0 and YEN = 0, all RRAMs that need to be set to HRS are selected and simultaneously initialized to HRS.

In traditional SRAM-based FPGAs, all LUTs must be

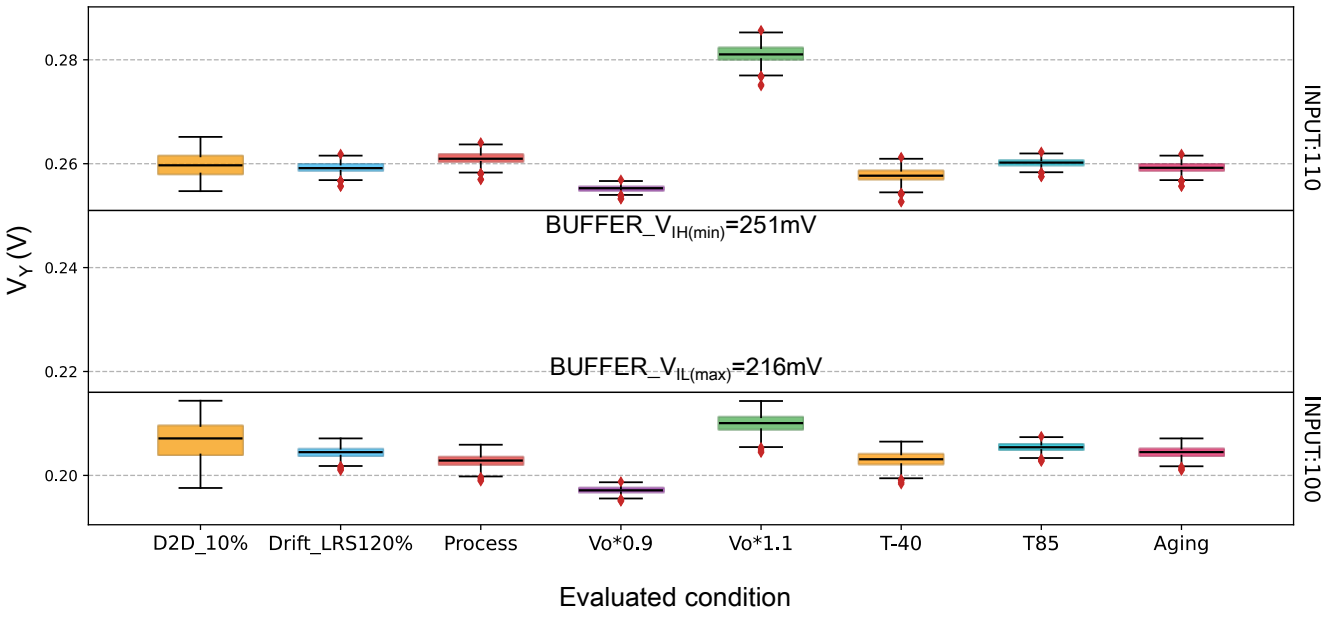


Fig. 7. Critical V_Y Voltages under device and circuit non-idealities.

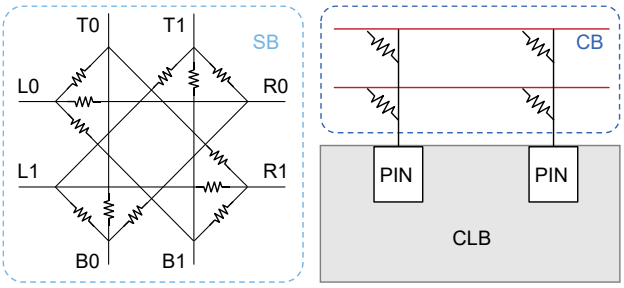


Fig. 8. Circuit structure of SB and CB

reprogrammed after every power-on. In contrast, the RCLU requires only a single initialization under ideal conditions. Even considering that RRAM has a limited retention time, re-initialization is only necessary when a significant shift occurs in the resistance values of RRAMs. Compared with traditional designs, the programming cost is still greatly reduced.

Nevertheless, although the RCLU itself does not require repeated SET/RESET operations during normal computation, FPGA-level reconfiguration still involves rewriting RRAM cells used for configuration and programmable interconnects. Therefore, the proposed architecture is more suitable for low-power IoT/edge applications with relatively stable configurations, rather than highly dynamic reconfiguration scenarios.

C. Reliability Evaluation of the RCLU

Fig. 7 shows the extracted V_Y distributions under representative device and circuit non-idealities. The upper group of box plots corresponds to input "110", while the lower group corresponds to input "100". The two horizontal solid lines indicate $V_{IL(max)}$ = 216mV and $V_{IH(min)}$ = 251mV (determined through simulation). Across all evaluated conditions, the V_Y distributions of input "110" remain above $V_{IH(min)}$, and the V_Y distributions of input "100" remain below $V_{IL(max)}$. This confirms that the output buffer can still distinguish the two critical logic states under the evaluated non-idealities.

The evaluated conditions include 10% RRAM device-to-device resistance variation, 20% LRS resistance increase for conductance-drift stress (a conservative drift-aware stress condition [38]), transistor process variation, ±10% operating-voltage fluctuation, temperature variation from -40°C to 85°C, and aging-induced transistor degradation (referring to [40], considering the dominant NBTI effect, the threshold voltage of PMOS increases by 50mV). Among these cases, the Vo*0.9 condition leads to relatively smaller separation from the buffer threshold, indicating that supply scaling is important boundary condition for the present RCLU design. In contrast, the Vo*1.1 condition increases the absolute V_Y levels and enlarges the separation for input "110", but the operating voltage still needs to be constrained to avoid unintended RRAM disturbance.

Cycle-to-cycle variation has a different implication for the proposed RCLU compared with stateful RRAM logic. In logic schemes that repeatedly switch RRAM devices during computation, the resistance state after each SET/RESET operation may vary from cycle to cycle and directly affect the next logic operation. In the proposed RCLU, the RRAM devices are initialized to predefined LRS or HRS states and are not repeatedly SET/RESET during normal computation. Therefore, cycle-to-cycle switching variation does not accumulate during RCLU logic evaluation. If significant conductance drift is observed and the programmed resistance states approach the design guardband, occasional verify-and-reinitialization can be applied to restore the predefined LRS/HRS states.

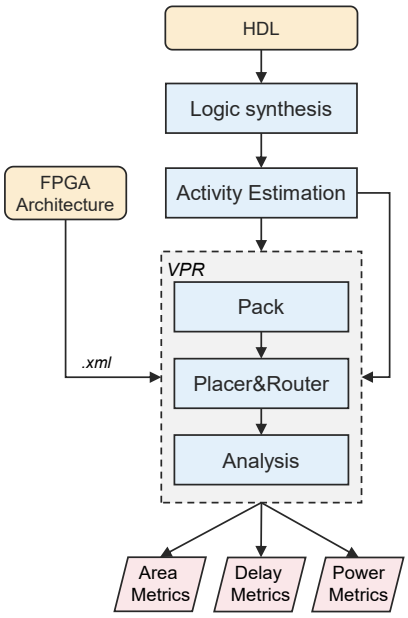


Fig. 9. Proposed CAD flow for area, delay and power analysis.

TABLE V
VPR ARCHITECTURE SETTINGS

Parameter	Definitions	Value
K	RCLU cell input size	2
N	Cluster size	20
I	CLB input size	22
O	CLB output size	20
L	Wire segment length	4
Fs	Switch block routing complexity	3
Fcin	Fraction of tracks connected to connection block	0.2
Fcout	Fraction of tracks a CLB output connects to	0.1

Overall, Fig. 7 demonstrates that the proposed RCLU maintains correct voltage-domain sensing under the evaluated RRAM variation, conductance drift, PVT fluctuation, and aging conditions. Nevertheless, the sensing margin is ultimately limited by the separation between the 1/2Vo and 2/3Vo voltage-division levels. Practical deployment therefore requires calibrated RRAM device statistics and combined variation analysis to determine the final design guardband.

D. Programmable Interconnect

For the programmable interconnect, we adopt the RRAM-switch-based design from [13], where RRAM cells are used to replace MUXs (Fig. 8). In this configuration, the HRS indicates disconnection, while the LRS indicates a conductive path—an approach that has been thoroughly evaluated in prior work.

IV. EVALUATION

In this section, we present the CAD flow for our RRAM-based FPGA described in section III. For comparison purposes, we select a classical and extensively used architecture provided by VTR as baseline SRAM-based FPGA [30]. To provide a complete comparison, we evaluate our work using the 20 largest MCNC benchmarks [28] and compare the results with recent RRAM-based and baseline SRAM-based FPGA designs [22].

A. CAD flow and Settings

To give a comprehensive analysis on performance and power of proposed architecture, the overall design flow is shown in Fig. 9. Benchmark circuits are first imported into Yosys for technology-independent logic optimization, and then tech-mapped into three standard logic gates (i.e., inv, nor2 and nand2) supported by RCLU. The ACE2 tool calculates the signal activity for power estimation. Finally, VPR performs placement and routing (P&R) process based on FPGA architecture description file. Estimates of area, delay and power metrics are then produced by the CAD flow based on P&R results.

In our designs, the low-power (LP) transistor model is adopted, and all the transistors are from the 40nm predictive technology model (PTM). The typical LRS and HRS of RRAM is set to 10KΩ and 100KΩ. In order to introduce complete parasitic effects in routing paths, a parasitic capacitance of 13.8 aF is added to RRAM model [23]. Based on the aforementioned RCLU design, the configurable logic block (CLB) in RRAM-based FPGA has 22 inputs, which contains 20 BLEs. In global routing networks, all routing tracks are built with length-4 wires ($L = 4$) to guarantee the best trade-off between short and long signal path [29]. The FPGA is set to minimum routable channel width to achieve the best performance. Detailed studies in [27] have shown that RRAM based multiplexer, of which the total capacitance is dominated by programming transistors, performs well in delay and power when using one-level structure. Note that the CB structure is not mentioned in [22]. Considering a relative low capacitance of an RRAM cell, we assume all the RRAM-based multiplexers adopt a one-level to provide desirable results. Parameter settings in VPR are shown in Table V.

B. Power-Delay Product

Compared to the traditional FPGA architecture (e.g., K6N10), the number of BLEs required in the proposed design to perform the same logic function is about 3-5 times. For instance, to implement alu4 benchmark, 2653 RCLUs are needed in our flow, while in comparison, 688 LUTs are needed for SRAM-based FPGA. This will lead to more complex routing wires of the global fabric. Therefore, we need to accurately estimate the RC value of the routing wires to obtain power and delay-related parameters. An RC scaling strategy is adopted to estimate the true RC parameters. Assume the ratio of conventional CLB area versus proposed CLB area is k , the corresponding resistance and capacitance

TABLE VI
COMPARISON BETWEEN DIFFERENT BASIC LOGIC UNITS

	Delay	Power
SRAM LUT [30]	139 ps	3.45 μW
RRAM LUT [22]	140 ps	3.37 μW
RCLU	97.6 ps	71.8 nW

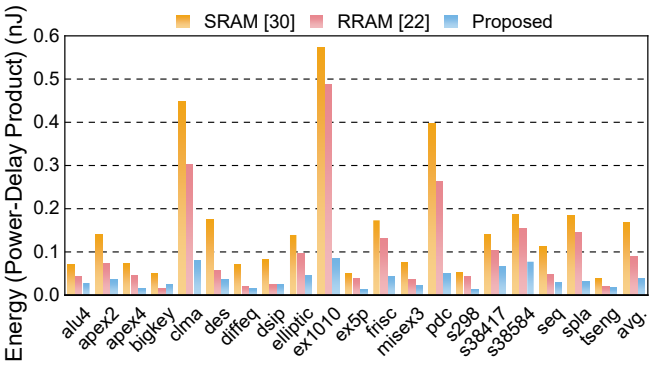


Fig. 10. Comparison of power-delay product between the proposed design and related work.

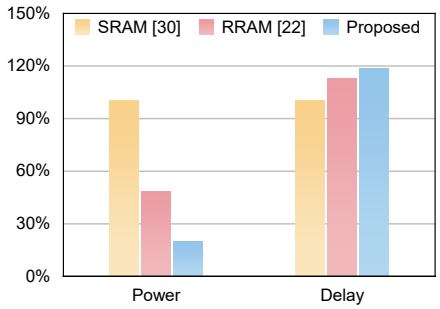


Fig. 11. Architecture-level normalized power and delay. comparison.

will be reduced by $\sqrt{k}$. Due to the area reduction achieved by the proposed CLB, the value of the exact RC parameters can be largely decreased (see Section IV-D).

Based on HSPICE simulation, the delay and power results of RRAM-based, SRAM-based LUT and proposed RCLU are reported in Table VI. As demonstrated in this table, the calculation delay of the proposed RCLU cell is 97.6 ps and its power consumption is 71.8 nW. All LUT designs exploit a tree-like structure with several level-restoring buffers in the signal path to prevent weak logic-1. The cell-level delay advantage of the RCLU stems from its fundamentally different signal path. Conventional LUT designs rely on a cascaded multiplexer tree, which requires multiple level-restoring buffers along the signal path to recover degraded logic levels and drive the output load. These buffers introduce significant delay and parasitic capacitance. In contrast, the RCLU performs voltage division directly within the resistive network and uses only a single sensing inverter followed by an output buffer to generate a full-swing digital output. The elimination of the multi-stage MUX tree and intermediate buffers results in both lower delay and lower power consumption at the cell level. The LUT design from [22] does not change the original structure of the SRAM LUT, only replacing SRAM cells with RRAM-resistor cells. Leakage power is effectively reduced by eliminating configuration cells, thus decreasing leakage paths from VDD to GND. Compared with SRAM LUT [30] and RRAM LUT [22], the proposed RCLU is 29.8% and 97.9% better in delay and power. The values are 30.3% and 97.8% smaller when compared with RRAM LUT. Compared to the SRAM LUT [30], the proposed RCLU exhibits 29.8% lower delay and 97.9% lower power consumption. When compared with the RRAM LUT [30], the delay and power are reduced by 30.3% and 97.8%, respectively.

Fig. 10 shows the power-delay product for SRAM-based, RRAM-based and proposed FPGA over 20 MCNC benchmarks. Experimental results reveal that the proposed design achieves an average PDP improvement of 76.89% and 57.09%. As detailed in the normalized breakdown of Fig. 11, this significant PDP reduction is primarily attributed to substantial power savings (51.71% and 80.25%, respectively). The power improvement is achieved by leveraging RCLU cells with low static power consumption (a major portion of the total power) and a compact CLB layout. The area reduction also decreases routing wire resistance and capacitance, leading to improved delay in the interconnect network. It should be clarified that Table VI reports the delay of a single basic logic unit at the cell level, while Fig. 11 presents the system-level delay of the full FPGA fabric after technology mapping, packing, placement, and routing. The finer granularity of the two-input RCLU means that benchmark circuits require more logic cells and longer routing paths compared with architectures using 6-input LUTs. This additional logic depth and routing complexity offset the cell-level delay advantage, resulting in slightly higher overall system-level delay. In terms of routing resource overhead, the proposed architecture requires 1.41× the minimum routable channel width compared with the baseline 6-input LUT architecture, due to the increased number of logic cells and interconnections. This routing overhead is the primary architectural cost of adopting the fine-grained, energy-efficient RCLU primitive. It is worth noting that the delay of the proposed design has only increased slightly by 18.4% and 5% despite the complex internal logic connections.

C. Programming Time and Energy

Several works have reported about the programming time and energy based on different device [18], [22], [26]. Programming energy of 1-bit RRAM is consumed by applying SET/RESET voltage between electrodes during write operation (~10ns), causing a relatively large power consumption of 0.1 pJ. In comparison, SRAM cell has 100× programming energy reduction (~1 fJ) in ps magnitude of time. [22] proposes a programming circuitry that can simultaneously SET/RESET the selected RRAM cells in a row during one programming cycle. The overhead of programming

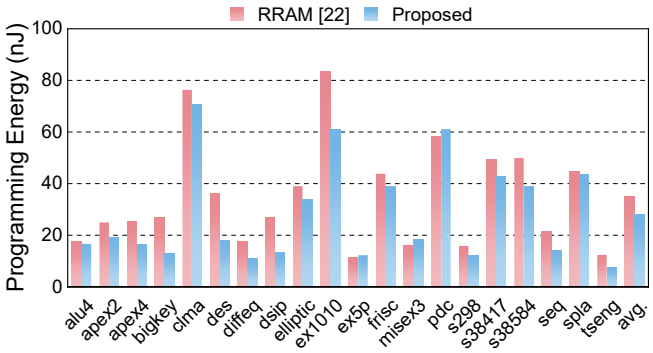


Fig. 12. Comparison of programming energy between the proposed design and related work.

TABLE VII

AREA PARAMETERS FOR DIFFERENT BASIC LOGIC UNITS

Parameter	Value	Parameter	Value
SRAM LUT [30]	932.1	SRAM-based CLB [30]	53894
RRAM LUT [22]	137.5	RRAM-based CLB [22]	2938.7
RCLU	26.5	Proposed CLB	2228.5

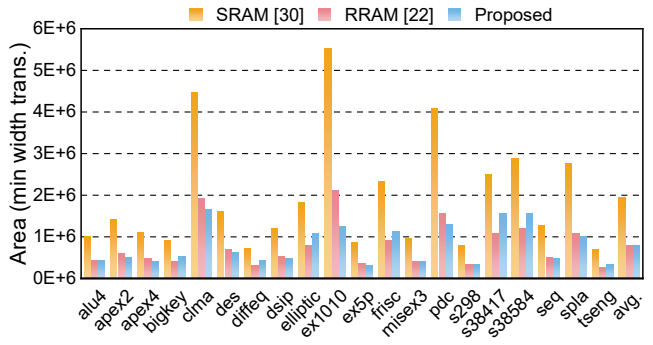


Fig. 13. Comparison of total area between the proposed design and related work.

an individual cell is largely compensated by this efficient scheme, which only leads to 81% configuration time increment (i.e., compared with SRAM-based FPGA).

Based on the programming energy of a single cell, we can simply estimate the programming energy by calculating the number of SRAM cells and RRAM cells. Given that SRAM has a significant advantage in programming energy, we evaluate the programming energy by comparing the proposed design with RRAM-based designs. It should be noted that in [22], the CB structure and corresponding programming steps haven't been considered yet. We assume one level RRAM-based multiplexer as the basic CB structure, which is the same as our work. As illustrated in Section III-B, RRAMs with LRS/HRS in the proposed RCLUs only need to be initialized once, thus no RRAMs need to be programmed. Modern FPGA prevails to adopt island-style architecture such as Xilinx Virtex Series, which is arranged in square array size of R × R. The number of RCLU cells in a CLB is denoted by N_{BLE} and W represents the channel width in routing network. The total number of RRAM cells needed to be programmed can be estimated according to the following equation:

$$N_{RRAM} = R^2 \times \frac{W}{2} \times 12 + W \times F_{cin} \times I \times \left(R^2 - R\right) \quad (1)$$

For baseline RRAM-based design, 2^K numbers of SRAM cells in each K-input LUT are replaced by RRAM-resistor cells that act as voltage-divider. Similarly, the total number of RRAM cells can be calculated according to equation (2).

$$N_{RRAM} = R^2 \times \frac{W}{2} \times 12 + R^2 \times N_{LUT} \times 2^K + W \times F_{cin} \times I \times \left(R^2 - R\right) \quad (2)$$

Fig. 12 compares the programming energy of the proposed design with baseline FPGAs. In the proposed architecture, the conventional LUT is replaced by the RCLU to implement combinational logic. Thus, the storage cells is largely reduced. In detail, RRAM-based FPGA [22] size for the pdc benchmark is 13 × 13 and the minimum routable channel width is 34, while the number of them for our design are 15 × 15 and 46, respectively. As mentioned in Section IV-B, 2-input RCLU deteriorates the complexity of routing networks. The programming energy for CB and SB increases by 57.86% compared with the baseline RRAM-based FPGA [22], while 100% reduction in programming energy of logic block fully compensates this increase. On average, a 19.3% reduction in programming energy is observed.

D. Area

The cell area of an RRAM is measured by $4F^2$ (F is the feature size) [18]. The size of output buffers in routing multiplexers are 5× minimum transistor width to provide drive strength. According to transistor sizing method in COFFE [25], we can give an accurate area estimation for our design in units of minimum-width transistor area (MWTA) by using two dimensionless equations. In detail, the equation (3) is to estimate the footprint of NMOS pass-transistor area and the equation (4) is for CMOS transistors area, in which x denotes the ratio of drive strength to minimum width transistor.

$$Area(x) = 0.447 + 0.128x + 0.391\sqrt{x} \quad (3)$$

$$Area(x) = 0.518 + 0.127x + 0.428\sqrt{x} \quad (4)$$

It should be noted that RRAMs can be stacked atop CMOS. The proposed RCLU cell consists of 4 RRAMs, 7 NMOS pass-transistors, 2 inverters and 1 buffer, resulting in a total area of 26.5 MWTA. As shown in Table VII, the area of baseline RRAM LUT is 548.1 MWTA. The area improvement of a single RCLU cell is 95.2% and 97.2%, compared with RRAM LUT and SRAM LUT, respectively.

Based on the structure of routing multiplexer and RCLU design, we compare the area of proposed FPGA with aforementioned studies in Fig. 13. Results reveal that only 1.06% improvement is observed when compared with RRAM-based FPGA [22]. It can be ascribed to two main reasons: (a) The reduced input size of logic primitives leads to a larger

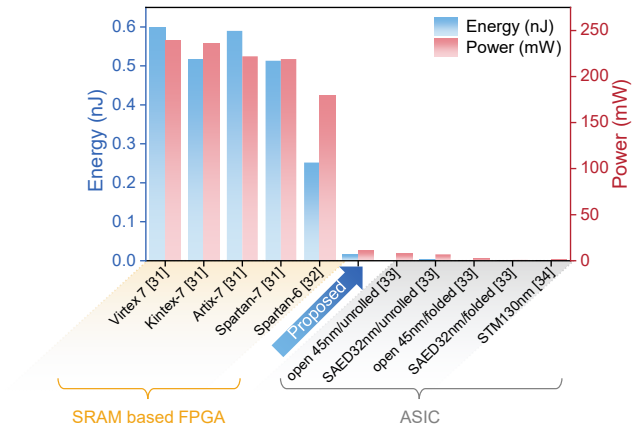


Fig. 14. Comparison of energy and power for running ASCON across different FPGA and ASIC architectures.

FPGA size, partially offsetting the area reduction; (b) Complex interconnection of netlist has largely increased the overall usage of routing elements.

V. APPLICATION

For comprehensive evaluation of the system's advantages in IoT applications, we benchmarked its performance running the ASCON lightweight encryption algorithm [16] against traditional FPGAs and ASICs [31]-[34], focusing on energy per bit and operating power (Fig. 14). Our architecture demonstrated significantly improved energy efficiency over conventional FPGAs, achieving an energy of 0.015884 nJ/bit (approximately 8.7 to 37.6 times lower than various reported FPGAs) and an operating power of 11.09 mW. This represents a substantial reduction from the tens to hundreds of milliwatts typically observed in SRAM based FPGA implementations for similar cryptographic tasks.

Compared to ASIC implementations, our RRAM-FPGA considerably narrows the energy-per-bit gap (0.015884 nJ/bit for our design versus a range of 0.000023 to 0.0025 nJ/bit for ASICs), although ASICs maintain an advantage in absolute energy efficiency. More critically, the 11.09 mW operating power of our RRAM based FPGA is comparable to that of several reported ASIC implementations of ASCON (e.g., those in the 6.8 mW to 7.97 mW range). This positions our reconfigurable RRAM-FPGA as an effective intermediate solution, achieving near-ASIC power characteristics.

In conclusion, our RCLU-based RRAM-FPGA successfully bridges the traditional performance gap between FPGAs and ASICs for lightweight cryptography. It mitigates the high energy consumption typically associated with FPGA-based solutions, achieving operating power levels comparable to some ASICs (11.09 mW), while preserving full hardware reconfigurability. This makes the proposed architecture a promising, energy-efficient, and flexible hardware solution for power-sensitive IoT applications, such as those found in smart healthcare and wearable technology.

VI. CONCLUSION

This paper presents an RCLU-based RRAM-FPGA architecture for low-power reconfigurable computing. Unlike conventional SRAM-LUTs and storage-replacement RRAM-LUTs, the proposed RCLU allows pre-initialized RRAM conductance states to directly participate in voltage-division-based logic evaluation, forming a localized logic-in-memory primitive at the FPGA logic-block level. Using a 2T2R structure, the RCLU performs voltage-domain sensing without RESET/SET operations, reducing latency and power while remaining compatible with FPGA routing and sequential circuits. Experimental results show that the proposed architecture offers substantial improvements in PDP, area, and programming energy compared to both traditional SRAM-FPGA and recent RRAM-based design. Moreover, our successful deployment of the ASCON lightweight encryption algorithm on the proposed FPGA highlights its suitability for IoT scenarios, demonstrating energy efficiency on par with ASIC implementations while preserving the flexibility of reconfigurable logic. Overall, this work provides a compelling pathway toward low-power, high-efficiency, and reconfigurable hardware platforms for next-generation IoT edge computing.

REFERENCES

[1] L. Ye *et al.*, “The Challenges and Emerging Technologies for Low-Power Artificial Intelligence IoT Systems,” *IEEE transactions on circuits and systems. I, Regular papers*, vol. 68, no. 12, pp. 4821–4834, 2021.

[2] M. Alioto, *Enabling the Internet of Things: From Integrated Circuits to Integrated Systems,* Cham, Switzerland: Springer, 2017.

[3] P. McEnroe, S. Wang and M. Liyanage, "A Survey on the Convergence of Edge Computing and AI for UAVs: Opportunities and Challenges," in *IEEE Internet of Things Journal*, vol. 9, no. 17, pp. 15435-15459, 1 Sept.1, 2022.

[4] M. Elnawawy, A. Farhan, A. A. Nabulsi, A. R. Al-Ali and A. Sagahyroon, "Role of FPGA in Internet of Things Applications," *2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, Ajman, United Arab Emirates, 2019, pp. 1-6.

[5] I. Kuon and J. Rose, "Measuring the Gap Between FPGAs and ASICs," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 2, pp. 203-215, Feb. 2007.

[6] Samir, Nagham, *et al.* “ASIC and FPGA Comparative Study for IoT Lightweight Hardware Security Algorithms.” *Journal of Circuits, Systems, and Computers*, vol. 28, no. 12, 2019.

[7] K. Jo, K. Cho and H. Yoon, "Variation-tolerant and low power look-up table (LUT) using spin-torque transfer magnetic RAM for non-volatile field programmable gate array (FPGA)," *2016 International SoC Design Conference (ISOCC)*, Jeju, Korea (South), 2016, pp. 101-102.

[8] K. Huang, Y. Ha, R. Zhao, A. Kumar and Y. Lian, "A Low Active Leakage and High Reliability Phase Change Memory (PCM) Based Non-Volatile FPGA Storage Element," in *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 9, pp. 2605-2613, Sept. 2014.

[9] S. Tanachutiwat, M. Liu and W. Wang, "FPGA Based on Integration of CMOS and RRAM," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 11, pp. 2023-2032, Nov. 2011.

[10] M. -F. Chang *et al.*, "A High-Speed 7.2-ns Read-Write Random Access 4-Mb Embedded Resistive RAM (ReRAM) Macro Using Process-Variation-Tolerant Current-Mode Read Schemes," in *IEEE Journal of Solid-State Circuits*, vol. 48, no. 3, pp. 878-891, March 2013.

[11] J. -M. Hung, X. Li, J. Wu and M. -F. Chang, "Challenges and Trends in Developing Nonvolatile Memory-Enabled Computing Chips for Intelligent Edge Devices," in *IEEE Transactions on Electron Devices*, vol. 67, no. 4, pp. 1444-1453, April 2020.

[12] X. Tang, G. Kim, P. -E. Gaillardon and G. De Micheli, "A Study on the Programming Structures for RRAM-Based FPGA Architectures,"

in *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 4, pp. 503-516, April 2016.
[13] J. Cong and B. Xiao, "FPGA-RPI: A Novel FPGA Architecture With RRAM-Based Programmable Interconnects," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 4, pp. 864-877, April 2014.
[14] S. Park, J. Kim and S. Yoon, "Energy-aware Placement for SRAM-NVM Hybrid FPGAs," *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Grenoble, France, 2020, pp. 858-863.
[15] S. Gupta, M. Imani and T. Rosing, "FELIX: Fast and Energy-Efficient Logic in Memory," *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, San Diego, CA, USA, 2018, pp. 1-7.
[16] Dobraunig, C., Eichlseder, M., Mendel, F. *et al.* ASCON v1.2: Lightweight Authenticated Encryption and Hashing. *J Cryptol* 34, 33, 2021.
[17] Fujiki, Daichi, et al. *In-/Near-Memory Computing*. 1st ed., Springer Nature, 2022.
[18] S. Yu and P. -Y. Chen, "Emerging Memory Technologies: Recent Trends and Prospects," in *IEEE Solid-State Circuits Magazine*, vol. 8, no. 2, pp. 43-56, Spring 2016.
[19] Wu, By Xin, Prabhuram Gopalan, and Greg Lara. *Xilinx Next Generation 28 Nm FPGA Technology Overview The Technology and Economic Challenge: Reducing Static Power to Increase Usable Performance And*. Vol. 312., 2011.
[20] P. -E. Gaillardon *et al.*, "Design and Architectural Assessment of 3-D Resistive Memory Technologies in FPGAs," in *IEEE Transactions on Nanotechnology*, vol. 12, no. 1, pp. 40-50, Jan. 2013.
[21] (2013). Predictive Technology Model (PTM). [Online]. Available: http://ptm.asu.edu/
[22] B. Khaleghi and H. Asadi, "A resistive RAM-based FPGA architecture equipped with efficient programming circuitry," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 7, pp. 2196-2209, Jan. 2018.
[23] Z. Wei *et al.*, "Distribution Projecting the Reliability for 40 nm ReRAM and Beyond based on Stochastic Differential Equation," in *Proc. IEEE International Electron Devices Meeting (IEDM)*, Dec. 2015, pp. 7.7.1-7.7.4.
[24] X. Tang *et al.*, "A RRAM-based FPGA for Energy-efficient Edge Computing," in *Proc. Design, Autom. Test Eur. Conf. Exhibit. (DATE)*, Mar. 2020, pp. 144-a.
[25] C. Chiasson and V. Betz, "COFFE: Fully-automated transistor sizing for FPGAs," in *Proc. Int. Conf. Field-Program. Technol. (FPT)*, Dec. 2013, pp. 34–41.
[26] X. Chen *et al.*, "Power and area efficient FPGA building blocks based on ferroelectric FETs," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 5, pp. 1780-1793, Oct. 2018.
[27] X. Tang, E. Giacomin, G. De Michieli and P.-E. Gaillardon, "Circuit designs of high-performance and low-power RRAM-based multiplexers based on 4T (ransistor) 1R (RAM) programming structure," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 64, no. 5, pp. 1173-1186, Dec. 2016.
[28] S. Yang, "Logic synthesis and optimization benchmarks user guide version 3.0," Microelectron. Center North Carolina, Research Triangle Park, NC, USA, Tech. Rep., Jan. 1991.
[29] X. Tang, G. De Micheli, and P.-E. Gaillardon, "A high-performance FPGA architecture using one-level RRAM-based multiplexers," *IEEE Trans. Emerg. Topics Comput.*, vol. 5, no. 2, pp. 210–222, Nov. 2016.
[30] K. Murray *et al.*, "VTR 8: High-performance CAD and customizable FPGA architecture modelling," *ACM Trans. Reconfigurable Technol. Syst. (TRETS)*, vol. 13, no. 2, May 2020, pp. 1-55.
[31] A. R. Alharbi, A. Aljaedi, A. Aljuhni, M. K. Alghuson, H. Aldawood and S. S. Jamal, "Evaluating Ascon Hardware on 7-Series FPGA Devices," in *IEEE Access*, vol. 12, pp. 149076-149089, 2024.
[32] S. Khan, W. -K. Lee and S. O. Hwang, "Scalable and Efficient Hardware Architectures for Authenticated Encryption in IoT Applications," in *IEEE Internet of Things Journal*, vol. 8, no. 14, pp. 11260-11275, 15 July15, 2021.
[33] Khan, Safiullah, et al. " Securing the IoT Ecosystem: ASIC-Based Hardware Realization of Ascon Lightweight Cipher. " *International Journal of Information Security*, vol. 23, no. 6, pp. 3653–3664, 2024.
[34] Kandi, Aneesh, et al. " Side-Channel and Fault Resistant ASCON Implementation: A Detailed Hardware Evaluation." *Proceedings / IEEE Computer Society Annual Symposium on VLSI*, 2024, pp. 307–312.
[35] C. Liu, et al. "Investigation on the Implementation of Stateful Minority Logic for Future In-Memory Computing," *IEEE Access*, vol. 9, pp. 168648-168655, 2021.
[36] X. Tang, E. Giacomin, G. De Micheli and P. -E. Gaillardon, "Post-P&R Performance and Power Analysis for RRAM-Based FPGAs," in *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 8, no. 3, pp. 639-650, Sept. 2018.
[37] C. Zambelli, M. Castellari, P. Olivo, and D. Bertozzi, "Correlating power efficiency and lifetime to programming strategies in RRAM-based FPGAs," in *Proc. New Gener. CAS (NGCAS)*, 2018, pp. 21-24.
[38] Rizzi, Tommaso, et al. "Process-Voltage-Temperature Variations Assessment in Energy-Aware Resistive RAM-Based FPGAs." *IEEE Transactions on Device and Materials Reliability*, vol. 23, no. 3, 2023, p. 1.
[39] Y. -C. Chen, W. Wang, H. Li and W. Zhang, "Non-volatile 3D stacking RRAM-based FPGA," *22nd International Conference on Field Programmable Logic and Applications (FPL)*, Oslo, Norway, 2012, pp. 367-372.
[40] T. Rizzi, et al. "Exploring the NBTI Aging and PVT effects on RRAM-based FPGA Multiplexers Performance," *2023 IEEE International Integrated Reliability Workshop (IIRW)*, South Lake Tahoe, CA, USA, 2023, pp. 1-5.

Xiaochang Zhang received the B.S. degree from Jilin University, Changchun, China, in 2020, and the M.S. degree from Peking University, Beijing, China, in 2023. He is currently pursuing the Ph.D. degree at the School of Integrated Circuits, Shanghai Jiao Tong University, Shanghai, China. His research interests include memory design, compute-in-memory architectures, emerging AI applications, and mixed-signal integrated circuit design.

Yongqiang Zhong received the B.S. degree in microelectronic engineering from the Ningbo University, Ningbo, China, in 2024. He is now pursuing the M.S. degree in integrated circuits engineering from Shanghai Jiao Tong University, Shanghai, China. His current research interests include reliability circuit design and electronic design automation.

Daoda An received the B.S. degree in Microelectronics Science and Engineering from Shanghai Jiao Tong University in 2025. He is currently pursuing the Ph.D. degree at Shanghai Jiao Tong University. His research interests include AI-assisted failure analysis and Design-Technology Co-Optimization (DTCO).

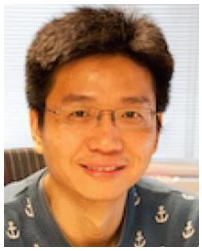
Yewei Zhang received her B.S. degree in Microelectronics Science and Engineering from East China Normal University in 2023. She is currently pursuing her Ph.D. in Integrated Circuit Science and Engineering at Shanghai Jiao Tong University. Her research interests include design-technology co-optimization (DTCO), process variation,

digital circuit reliability, and silicon lifecycle management (SLM).

Pengpeng Ren received the Ph.D. degree from Peking University, Beijing, China, in 2016. After graduation, he worked in HiSilicon Technologies as a staff engineer. He is currently an Associate Professor with the department of Micro/Nanoelectronics, Shanghai Jiao Tong University since 2021. His current research interests include nanoscale CMOS reliability physics, design for reliability, and AI for EDA. He has authored or co-authored over 70 scientific papers, including papers published in IEDM, VLSI-T, IRPS, IEEE TCAD and TED.

Runsheng Wang received the B.S. and Ph.D. degrees from Peking University in 2005 and 2010, respectively. He was a Visiting Scholar at Purdue University (2008–2009). He is currently a Full Professor with the School of Integrated Circuits and Associate Dean of the School of EECS, Peking University. He has (co-) authored over 190 articles, including 40+ in IEDM and VLSI Technology, and holds over 50 patents worldwide. His research interests include nanoscale CMOS, reliability, DTCO/EDA, and emerging computing technologies.

Yimao Cai (Member, IEEE) received the B.Eng. degree from Beijing Normal University, Beijing, China, in 2000, the Ph.D. degree from Peking University, Beijing, in 2006, and the M.S. degree from Beijing Normal University in 2013. He is currently a Professor with the School of Integrated Circuits, Peking University. His research interests include advanced memory device, neuromorphic device, and chip design.



Feng Hao (Senior Member, IEEE) received the PhD degree in computer science from the University of Cambridge, in 2007. After working in security industry for several years, he joined as a lecturer with the School of Computing, New castle University, in 2010, where he became a reader, in 2014 and a professor, in 2018. Currently, he is a professor in security engineering with the Department of Computer Science, University of Warwick. His research interests include applied cryptography, system security, and efficient computing algorithms.

Zhigang Ji (Member, IEEE) received the B.Eng. degree in electrical engineering from Tsinghua University, Beijing, China, in 2003, the M.Eng. degree in microelectronics from Peking University, Beijing, in 2006, and the Ph.D. degree in microelectronics from Liverpool John Moores University, Liverpool, U.K., in 2010. He was a Reader with Liverpool John Moores University. He is currently a Professor with the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China. He has authored or co-authored over 100 journal articles. His research interests include micro/nano electronic devices, reliability physics, circuit/device co-design for reliability, novel paradigm computing, and hardware security.