Dependability of Future Edge-AI Processors

Pandora's Box

Gomony, Manil Dev; Gebregiorgis, Anteneh; Fieback, Moritz; Geilen, Marc; Stuijk, Sander; Richter-Brockmann, Jan ; Bishnoi, Rajendra; Taouil, Mottaqiallah; Hamdioui, Said; More Authors

DOI
10.1109/ETS56758.2023.10174180

Publication date
2023

Document Version
Final published version

Published in
Proceedings of the 2023 IEEE European Test Symposium (ETS)

2023 28th IEEE European Test Symposium (ETS)
- European Project -

Dependability of Future Edge-AI Processors: Pandora's Box

Manil Dev Gomony[1], Anteneh Gebregiorgis[2], Moritz Fieback[2], Marc Geilen[1], Sander Stuijk[1], Jan Richter-Brockmann[3], Rajendra Bishnoi[2], Sven Argo[3], Lara Arche Andradas[4], Tim Güneysu[3], Mottaqiallah Taouil[2], Henk Corporaal[1], and Said Hamdioui[2]

[1]Eindhoven University of Technology, [2]Delft University of Technology, [3]Ruhr-Universität Bochum, [4]Thales Alenia Space

2023 IEEE European Test Symposium (ETS) | 979-8-3503-3634-4/23/$31.00 ©2023 IEEE | DOI: 10.1109/ETS56758.2023.10174180

***Abstract*—This paper addresses one of the directions of the HORIZON EU CONVOLVE project being dependability of smart edge processors based on computation-in-memory and emerging memristor devices such as RRAM. It discusses how how this alternative computing paradigm will change the way we used to do manufacturing test. In addition, it describes how these emerging devices inherently suffering from many non-idealities are calling for new solutions in order to ensure accurate and reliable edge computing. Moreover, the paper also covers the security aspects for future edge processors and shows the challenges and the future directions.**

***Index Terms*—ULP, dynamic DL, edge-AI, SoC, memristor, approximate computing, DSE, compiler stack.**

I. Introduction

Artificial Intelligence (AI) has evolved into a technology for a wide range of purposes; it impacts economy and society in an accelerated manner. Furthermore, the IoT-edge partnership is expected to revolutionize data computing. Today's AI solutions mainly target cloud and datacenters; they are expensive, power hungry, consume lots of silicon area, and off-chip memory burns power and taxes memory bandwidth. This makes them unsuitable for energy constrained edge applications. Hence, both academia and industry have been investigating and exploring new energy efficient smart edge computing engines in the light of traditional scaled CMOS technologies as well as of emerging post-CMOS device technologies, while incorporating radically new brain-inspired concepts [1], [2]. Although all of these seem to be extremely promising, these alternative, energy-efficient computing paradigms are worthless if the *hardware dependability* is not ensured, which includes *quality*, *reliability*, and *security*. Quality is defined as the ability of the computing engine to perform its function according the specification at time zero, i.e., right after its production, hence ensuring the detection of all manufacturing defects through appropriate tests [3], [4]. Reliability is defined as the ability to perform the function for the lifetime of the targeted application; hence ensuring the correct function in the presence of; e.g., ageing defects is essential [5], [6]. Security is defined as the ability to ensure resiliency against any attack that could target data, IP (intellectual property) or alternation of the function [7].

Although hardware dependability of CMOS-based computing engines is quite established, it is still in its infancy stage when it comes to non von-Neumann architectures, such as Compute-in-Memory (CIM) and post-CMOS device-based (e.g., memristor-based) computing [8]–[10]. Such emerging architectures and devices are not only giving rise to new and unique defect and failure mechanism during their manufacturing, but they also suffer (inherently) from many non-idealities. Non-idealities consists of internal factors (e.g., drift of components) causing intermittent faults or permanent faults, and external factors (e.g., slow changes in its environment) causing transient faults. Moreover, one of the major drawbacks of using memristor computing for edge AI is the serious theft threats they face; e.g., well-trained neural network models are seen as intellectual property and, when loaded in the memristor computing systems at the edge, they may be easily stolen if no special countermeasures are put in place.

This paper addresses one of the directions the CONVOLVE (Smart and seamless design of smart edge processors) project [11] covers; i.e., hardware dependability for edge AI processors based on emerging device technology. The paper discusses the three mains aspects of hardware dependability: quality, reliability, and security. It highlights the challenges, the state of the art and further research directions. In summary, the contributions of the paper are:

- Quality: it shows how alternative computing paradigms (such as CIM) based on emerging devices is changing the way we used to perform manufacturing test. It also illustrates how new defects mechanisms are involved, which calls for new ways of defect modeling and testing.
- Reliability: it highlights the challenge computing-in-memory faces from non-idealities of emerging devices, and presents different solutions to deal with them. It discusses the non-ideality challenges that need to be addressed for reliable operations and illustrates the directions addressing those issues at different levels of abstraction.
- Security: it highlights the different adversarial models and the challenges in providing long-term Security in emerging computing paradigms such as compute-in-memory and neuromorphic computing. In addition, the challenges in incorporating Post Quantum Crypto security and Trusted Execution Environments (TEE) in edge-AI processors are discussed.

The rest of the paper is organized as follows. Section II briefly gives and overview on CONVOLVE and its main objectives. Section III covers Quality, Section VI reliability and Section V security. Section VI concludes the paper.

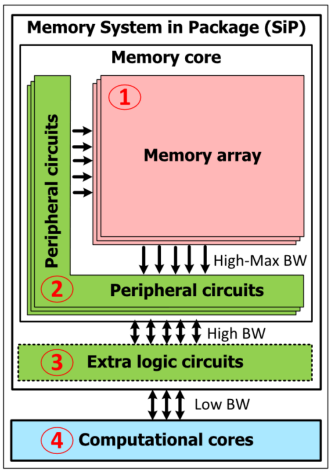

Figure 1. Classification of computer architectures [12]

II. CONVOLVE Overview

Computer architectures can be classified based on where the computation takes place with respect to where the data are stored. Fig. 1 shows that there are four classes [12]. Classes 1 and 2 are Compute-in-Memory (CIM) implementation, as they perform the computation in the memory core itself; class 3 is a Compute-Near Memory (CNM) implementation, as it performs computation directly next to the memory; finally, class 4 is a traditional Von Neumann computer architecture where the computation is performed in a separate processor that is connected to the memory. The higher the class number, the lower the bandwidth (BW) is to perform computations and the more costly it is to move data for computations. The CIM classes can be further divided based on where the result of the computation is generated: In class 1, the result is generated within the memory array (A) itself, called CIM-A; while in class 2 the result is generated in the peripheral (P) circuits, e.g., by using dedicated sense amplifiers (SAs), called CIM-P. An example of CIM-A is Majority logic [13], and an example of CIM-P is Scouting logic [14].

CONVOLVE project [11] aims at developing a novel framework consisting of hardware and software that allows anyone to deploy a custom designed secure and reliable SoC with 100X energy efficiency for edge AI applications in 10X less design time. This involves the development of an ultra-low-power (ULP) library with novel (CIM) - both digital (SRAM based) and analog (RRAM based), and a flexible processing array with CNM architecture keeping processing very close to the memory to reduce energy consumption. The ULP blocks will be designed with common or standard interfaces, and optimized at micro-architecture, circuit and device levels. The accelerator blocks are optimized to execute the computation patterns of both Artificial Neural Networks (ANN) and Spiking Neural Networks (SNN) efficiently.

CONVOLVE researches efficient design-space exploration (DSE) techniques combining different levels of hierarchy in a compositional way, i.e., hardware and software components can be seamlessly glued together, while guaranteeing overall behaviour and reliability; this deals with the SoC heterogeneity and supports efficient mapping of applications to hardware architectures consisting of CIM-based accelerators. An SoC architecture will be generated using these modular architecture templates after performing automated DSE; this allows the evaluation of all architecture possibilities while considering the dependability requirements. Since CIM based architectures are far less reliable than traditional CMOS circuits, novel mechanisms will be devised to deal with hardware variability. The techniques will include increasing the robustness by dynamically adapting computational precision, data path width, early-termination, skipping layers/neurons, etc. at different levels of design hierarchy.

CONVOLVE aims to provide hardware security against known attacks and real-time guarantees by compositional implementation of Post Quantum Cryptography (PQC) and real-time Trusted Execution Environment (TEE). PQC accelerator blocks will be designed with standard interfaces that can be plugged into a modular architecture template to make hardware secure, even in the long term (over a decade). Furthermore, CONVOLVE develops design-for-security methodology and makes sure that all security features can be added in a compositional manner while providing real-time guarantees. We will explore design for robustness, to deal with in-field failures and non-ideal real-world environments.

CONVOLVE considers four different use-cases with diverse dependability requirements: (1) On-board computer vision in Earth Observation (EO) satellites (2) Deep Noise Suppression and Spech Quality Prediction in audio devices, (3) Acoustic Scene Analysis in automotive systems, and (4) Video-based Traffic Analysis for traffic monitoring systems. The microprocessors in these systems needs to be designed for robustness against environmental changes and security attacks. For e.g. secure firmware updates refer to the process of securely and trust-worthily updating the software controlling the hardware components of an (edge) device, such as a smartphone, IoT device or headset. This is important because vulnerabilities or bugs in the firmware could potentially be exploited by attackers to gain unauthorised access or cause damage to the (edge) device, and to ensure that the device remains secure and up-to-date.

III. Testing

Emerging memories based CIM-based accelerators can operate in two different *configurations*: as a regular memory in the *Memory Configuration (MC)*, or as a computing device in the *Computation Configuration (CC)*. To operate in the CC, design changes to existing memories need to be made, e.g., in CIM-A, the array needs to be modified so that the compute results can be written back, while in CIM-P the peripheral circuits needs to be modified to perform computations. These adapted circuits introduce new faults that are not seen in regular memories [3], [15], [16]. For example, Scouting logic [14] based CIM-P (which supports the execution of logic operations) requires the modification of the sense amplifier to ensure the comparison with multiple references, and the redesign of the address decoders to ensure the selection of multiple operands (i.e., rows) simultaneously. Obviously, not all circuits used in the MC are used in the CC, and vice-versa. Therefore, regular memory tests cannot guarantee the detection of all CIM faults. Instead, to guarantee high product quality, CIM-based accelerators needs to be tested in *both*

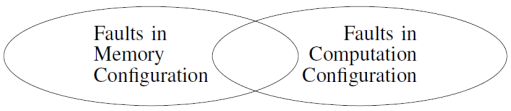

Figure 2. Faults in MC and CC [15]

configurations: MC and CC [3], [15], [16]. Next, we discuss testing of these two configurations for a RRAM-based CIM implementation supporting Scouting logic [14].

A. Testing in the Memory Configuration

In the memory configuration, the CIM circuit works as a regular memory. As such, in this configuration, defects will sensitize the same faults in the cell array as in regular memories. RRAM devices suffer from manufacturing defects that result in faults that are not observed in traditional memories [17]. Furthermore, because a RRAM device is a non-linear device, these defects cannot be modelled using the traditional defect modeling approaches that use linear resistors and doing so will to test escapes [17]. Therefore, appropriate defect modeling approaches that accurately incorporate the defective behavior in the defect model are needed, such as the Device-Aware Test (DAT) approach [18]. DAT was successfully deployed to accurately model and detect unique defects in RRAM memory cell array of a CIM-P supporting scouting logic; examples of such defects/faults are forming defects and intermittent undefined state faults [3], [15], [19]. Moreover, testing CIM in the MC is more than just testing the memory array. Dedicated test solutions are required to deal with defects in the peripheral circuits, such as address decoders, sense amplifiers, drivers, etc.

B. Testing in the Computation Configuration

In the computation configuration, the CIM circuit works as a computation device. For Scouting logic with two operands this means that two cells can be addressed at once, requiring an additional address decoder, and that the resulting current is compared against an appropriate reference that depends on the selected logic operation. Due to this, different faults are sensitized in the CC for the array, the address decoders, and the sense amplifiers. In the array, compute faults are sensitized, e.g., an OR operation may cause a cell to flip its state [20]. In the address decoders, port interference faults can be sensitized, e.g., the address in one decoder influences the address in the other decoder [3]. In the sense amplifiers, similar faults as in the MC are sensitized, but now also for the additional reference, e.g., in the CC the sense amplifier may be slow for an OR operation, but not for an AND operation [3].

The fault analysis results show that using CIM in the CC gives rise to unique faults that requires special operations to detect them. Hence, faults sensitized in the CC are not a subset of faults in the MC, nor are they a superset, as shown in Fig. 2 [15]. For example, there exist bridging defects that only sensitize a fault in the CC. Besides exclusive faults, there are also many defects that sensitize faults in both configurations. This gives an opportunity to optimize a test solution so that a maximal number of defects can be detected with a minimum of sensitizing sequences. From the above, it follows that a CIM-based accelerators need to be tested in *both configurations* to achieve the highest quality possible, but that testing in both configurations also allows to optimize test solutions.

C. Directions

Research and development for CIM-based accelerators that make use of emerging devices is still in its infancy stage and many open questions need to be worked out.

Other architectures and kernels: It is expected that the above procedure of defining the architectural differences also apply when testing other CIM architectures and kernels, such as matrix-vector multiplication accelerators, or arithmetic CIM. In these applications, more hardware changes are made with respect to a regular memory, e.g., through the addition of analog-to-digital converters. Hence, it can be expected that even more unique faults will be identified and that new test solutions need to be developed to detect them.

Emerging memories: CIM-based accelerators can also be made using other emerging memory technologies, such as STT-MRAM and PCM. These are also susceptible to novel defects and thus require proper defect modeling in order to develop high-quality tests [18]. Furthermore, when used in a CIM-based accelerator, more novel faults may be observed. These need to be detected with appropriate test solutions, as well.

Optimizations: Compute operations can be used to speed up testing and increase the detection capabilities. For example, in [21], compute operations are used to detect unique RRAM faults in a more efficient manner than could be achieved using only memory operations. As such, the usage of CIM also gives opportunities to increase test efficiency and fault coverage.

IV. Reliability

Memristors suffer from several non-idealities where some occur at the production time before shipping and others, during the product life cycle. Thus, they are classified as time-zero and time-dependent non-idealities, respectively.

Time-zero non-idealities [21]–[24]: these consist of: a) *Variation* (Variation is the deviation of the memristor resistance value after programming from the expected value, due to fabrication imperfections.), b) *Wire parasitics* (Due to the finite parasitic resistance and capacitance of the interconnect wires, signals suffer from delay mismatch and voltage degradation.), c) *Non-zero G_{min} error* (Non-zero G_{min} error occurs when a non-zero output current is produced by applying a non-zero input voltage to a memristor with G_{min} conductance), etc.

Time-dependent non-idealities [20], [25]–[27]: these consist of: a) *Endurance* (Memristors suffer from limited endurance due to the destructive nature of the programming operations.), b) *Device degradation* (Due to stress and ageing, CMOS periphery and memristors in CIM suffer from device degradation), c) *Conductance drift* (The conductance states of the memristors drifts with time and can eventually lead to unwanted bit-flips), d) *Read disturb* (Read disturb is a phenomenon where a correct value is returned during a read operation, while the stored value is flipped by the read operation), etc.

Next, we will highlight some solutions to mitigate the impact of non-ideality on RRAM-based CIM, and thereafter briefly describe some further directions.

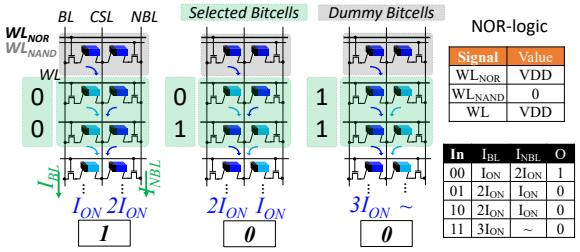

Figure 3. Complementary 2T2R bitcell configuration for NOR operation [30].

A. Potential solutions

Solutions for mitigating the impact of non-idealities can be developed at various abstraction levels such as circuit-level [28], architecture-level [29] and application-level [5]. Next two solution examples will be illustrated.

1) Referencing-in-Array scheme: This circuit level scheme aims at using 2T2R cell structure rather than 1T1R to create a complementary bitcell structure that inherently acts also as a reference during the operation execution [30]; this results in a high sensing margin. The higher sensing margin of this scheme can effectively mitigate the impact of process variation and wire parasitics and increase the number of operands upto 56 in a single cycle. Figure 3 shows the 2T2R-based cell structure and configuration that can be used for NOR and AND bit-wise operations; a dummy row is an integral part of such structure (e.g., top row). Each dummy cell is pre-programmed, while each 2T2R cell presents one bit; the data-bit '1' is represented by dark blue (LRS) RRAM device and the complementary data-bit as light blue (HRS). Pass transistors are use to connect these memristors to bitline (BL) and negative bitline (NBL). Dummy cells have two two independent WLs, namely WL_{NOR} (selected for NOR operation) and WL_{NAND} (selected for NAND operation), connected to the BL and NBL-sided pass transistors, respectively. The two memristor devices of each dummy cell is set to LRS. The right side of Figure 3 illustrates NOR logic operation; two rows storing operands are activated simultaneously along with WL_{NOR} of the dummy row. This results not only in accurate and and reliable NOR operations as the sensing margin is now higher, but also in an energy efficient operation (more than 11X as compared to state-of-the-art 1T1R based solutions).

2) Unbalanced bit-slicing scheme: This architecture level scheme aims to employ an unbalanced bit-slicing (UBS) scheme in [31], which changes the way neural weights are mapped into a CIM crossbar in order to mitigate the impact of non-zero G_{min} error. The UBS scheme splits the bits of neural weights into unbalanced slices and different number of RRAM devices are allocated per slices in order to minimize the impact of non-zero G_{min} error and improve the sensing margin. Thus, the UBS scheme allocates a one-bit RRAM for the MSB slices and multi-bit (e.g., 2-bit) RRAM for the LSB slices. This provides sufficient sensing margin to the MSB column output and make them immune to non-zero G_{min} error. Moreover, the USB scheme uses of 2's complement arithmetic to further reduce the impact of accumulative non-zero G_{min} error after combining the partial outputs due to weighted subtraction as shown in Eq. 1a and Eq. 1b.

$$E_f = 64{\cdot}E_1 + 16{\cdot}E_2 + 4{\cdot}E_3 + E_4 \quad (1a)$$

$$E_f = (-128){\cdot}E_1 + 64{\cdot}E_2 + 16{\cdot}E_3 + 4{\cdot}E_4 + E_5 \quad (1b)$$

The UBS scheme can effectively mitigate the impact of non-zero G_{min} error and achieve up to 8.8× and 1.8× classification accuracy compared to state-of-the-art CIM architectures for single-bit memristors and two-bit memristors respectively, at reasonable energy overheads.

B. Directions

In order to deal with non-idealities for a complete system, it is important to provide solutions at all abstraction layers. Following are the potential solutions:

Device level solutions involve improving the device structure and material composition for better characteristics to ensure well-defined distinctive resistance states with low variation.

Circuit level solutions include innovative circuit designs to improve accuracy in the presence of non-idealities. This involves the design space exploration of several bitcell configurations, dedicated referencing schemes, and variation-aware ADC and DAC converters.

Architecture level solutions involve mitigate the impact of non-idealities by changing the way in which an application is mapped to CIM hardware, changing how the data flows through various CIM system components or introducing error correction mechanisms.

Application level solutions involve adapting the underlying applications so that inexact computations will suffice instead of exact ones. Applications that require one or more static-valued operands also are well suited for memristor-based CIM as endurance problem is alleviated.

V. Security

CONVOLVE aims at providing high security guarantees to edge-AI processors with CIM based accelerators against a wide range of powerful adversary models precisely fitted to its application domain. This covers current state-of-the-art adversaries but also potential future attackers to ensure its longevity and endurance even when confronted with new technologies. To this end, we first introduce our adversary models, followed by presenting four main research challenges which should be addressed by CONVOLVE.

A. Adversary Model

CONVOLVE considers two categories of adversary models based on their level of access to the device: Adversaries restricted to digital access and with physical access.

Adversaries restricted to digital access: In the first adversary model, we assume that the access to the device is constrained to the provided communication streams of the target, i.e., remote (or local) connections to digital data channels or memories. An attacker can record, filter or manipulate the incoming and outgoing data or communicate with the device. This also includes having remote access to the same target computing unit which, for example, allows the attacker to measure *runtime differences* [32] or mount more advanced attacks like *Rowhammer* [33]. Moreover, we assume that an adversary is capable of recording and storing encrypted communications. More precisely, data that is today encrypted by common asymmetric schemes could eventually be decrypted by *large-scale quantum computers*. One prominent approach for this purpose is Shor's algorithm [34].

Adversaries with Physical Access: The second adversary model considers advanced attackers with local, physical access to the device. In addition to the input-output behavior of the device and regular communication channels, the adversary can use additional physical information about the target device, e.g., its power consumption [35] or electromagnetic radiation [36]. More precisely, the *power consumption* of edge-AI processors highly depends on the processed data. Hence, in the context of cryptographic implementations, an adversary can extract information about the secret key material from power traces acquired during an encryption or decryption process. While these power side-channel attacks are seen as passive attacks, *fault-injection attacks* are considered as active attacks and aim to alter an intermediate state of a cryptographic algorithm [37]. The adversary exploits the faulty output in order to extract information about the secret key material.

B. Challenges

Within CONVOLVE, we identify four main challenges with respect to the security of the envisioned architecture.

Long-term Security: Data that is processed today may be sensitive for decades to come. Consequently, the data must be secured not only against current but also against potential future adversaries, such as quantum computers Section V-A. The strong cryptographic algorithms must provide long-term security while ensuring alignment with CONVOLVE's application domain. Furthermore, the target algorithms need to run on devices with limited resources such as power and memory.

Security for CIM architectures: As Computing-in-Memory (CIM) devices become a reality, numerous threats arise with them. CIM devices are vulnerable to several types of attacks including fault-injection and side-channel attacks. The literature cites examples of each type of attack. Flex et al. present a fault-injection attack on logic-in-memory (a subset of CIM) neuromorphic hardware used as BNN accelerator to demonstrate misclassification in MNIST dataset [38]. In another instance, Ziyu et al. present a side-channel attack to reverse engineer the deep neural network (DNN) implemented on a simulated CIM architecture [39].

Security Aware Design-Flow: Current design flows for software and hardware implementations are optimized to generate fast, small, or energy-efficient designs. However, when implementing cryptographic algorithms for software and hardware, we often introduce additional redundancy to avoid timing or power side channels. The challenges manifest slightly differently in software-based and hardware-based design flows. In the former, we need to avoid any runtime differences based on confidential data which can, for instance, be introduced by secret-dependent branching or memory accesses. In the latter, the most significant challenges lie in preventing physical attacks as described in Section V-A for the adversary with physical access to the device.

Modern TEE: Trusted Execution Environment (TEE) aim to provide additional security by isolating high-risk software from untrusted code. An example for such high-risk software could be the cryptographic components that ensure the confidentiality and integrity of an ML model. A typical Operating System (OS) in this example would be untrusted because OSs are complex, often containing millions of lines of code that are almost certain to contain bugs. Thanks to the hardware enforced isolation, a TEE would ensure that even the OS could not access the secrets inside the TEE [40], [41].

C. Future Directions

Long-term Security: The de-facto standard countermeasure against emerging quantum computers is *post-quantum cryptography*. It provides strong formal security guarantees and can be coupled with "classic cryptography" to further enhance reliability and security [42]. Also, efficient, timing side-channel resistant implementations are available for practical applications [43]. However, resistance against power side channels or fault injections is more challenging to achieve, particularly in combination with high-performance or low-power requirements [44]. Within CONVOLVE, the combination of strong security and low-resource consumption, as well as the modularity and composability required for the 10x faster development process, will be the focus of research.

Security for CIM architectures: The threats can be minimized by adding data authentication mechanisms and embedding effective countermeasures. Similar to long-term security, the additional latency, power, or chip area overheads required to implement appropriate countermeasures must be carefully evaluated and minimized without compromising security.

Security Aware Design-Flow: Within CONVOLVE, we aim for novel design-flow architectures that inherently consider security aspects in the early design process. For software-based design flows, any constant-time execution guarantees with respect to secret data should be automatically detected [45] and maintained. For hardware-based design flows, physical attacks are the predominant problem to be addressed. Common countermeasures against side-channel attacks are based on the secret sharing approach [46], while countermeasures against fault-injection attacks often utilize mechanisms introducing redundancy in time or area [47]. To this end, modern security-aware synthesizers should recognize parts of the hardware description belonging to countermeasures and preserve their security requirements and underlying security assumptions for predefined models [48], [49].

Modern TEE: Several implementations of TEE already exist, such as Intel SGX [50] and ARM TrustZone [51]. However, many of these implementations have been attacked over the years, due to e.g., implementation bugs or side-channel attacks [52]. Open-source frameworks such as the Keystone project aim to provide secure, reliable and yet still flexible TEE implementations for platforms such as RISC-V edge processors [40].

VI. Summary

This paper briefly addressed one of the key aspects of future edge-AI processors based on emerging devices (RRAM as an example). It is clear that solving the dependability challenges to ensure high quality, reliable and secure computing engines is crucial and critical for the deployment of such devices in real applications. Computation-in-memory based on emerging devices uses new device technologies, requires new designs, modified system architecture, disparate programming models, etc; all of these are fundamentally changing the assumptions,

the models and solutions with regards to quality, reliability and security as compared with traditional computing.

ACKNOWLEDGMENT

This work is funded by EU's Horizon Europe research and innovation programme under grant agreement No. 101070374.

REFERENCES

[1] A. Lines *et al.*, "Loihi asynchronous neuromorphic research chip," in *IEEE ASYNC*, 2018.
[2] C.-X. Xue *et al.*, "A cmos-integrated compute-in-memory macro based on resistive random-access memory for ai edge devices," *Nature Electronics*, 2021.
[3] S. Hamdioui *et al.*, "Testing computation-in-memory architectures based on emerging memories," in *2019 IEEE International Test Conference (ITC)*, 2019, pp. 1–10.
[4] E. I. Vatajelu *et al.*, "Challenges and solutions in emerging memory testing," in *IEEE Transactions on Emerging Topics in Computing*, vol. 7(3), 2017, pp. 493–506.
[5] A. Gebregiorgis *et al.*, "Dealing with Non-Idealities in Memristor Based Computation-In-Memory Designs," in *IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, 2022, pp. 1–6.
[6] J.-F. Li, "Testing and reliability of computing-in memories: Solutions and challenges," in *IEEE International Test Conference in Asia*, 2022.
[7] M. Tehranipoor *et al.*, *Introduction to Hardware Security and Trust*. Springer Publishing Company, Incorporated, 2011.
[8] M. Zhao *et al.*, "Reliability of analog resistive switching memory for neuromorphic computing," in *DATE*, 2015.
[9] S. Hamdioui *et al.*, "Testing computation-in-memory architectures based on emerging memories," in *ITC*, 2019.
[10] M. Zou *et al.*, "Review of security techniques for memristor computing systems," in *Frontiers in Electronic Materials*, 2022.
[11] M. D. Gomony *et al.*, "Convolve: Smart and seamless design of smart edge processors," in *DATE*, 2023.
[12] H. A. D. Nguyen *et al.*, "A classification of memory-centric computing," *J. Emerg. Technol. Comput. Syst.*, vol. 16, no. 2, jan 2020.
[13] E. Linn *et al.*, "Beyond von neumann—logic operations in passive crossbar arrays alongside memory operations," *Nanotechnology*, vol. 23, no. 30, p. 305205, jul 2012.
[14] L. Xie *et al.*, "Scouting logic: A novel memristor-based logic design for resistive computing," in *2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2017, pp. 176–181.
[15] M. Fieback *et al.*, "Testing scouting logic-based computation-in-memory architectures," in *IEEE European Test Symposium (ETS)*, 2020, pp. 1–6.
[16] T.-L. Tsai *et al.*, "Testing of in-memory-computing 8T SRAMs," in *2019 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*. IEEE, 2019, pp. 1–4.
[17] M. Fieback *et al.*, "Testing resistive memories: Where are we and what is missing?" in *ITC*, 2018.
[18] ——, "Device-aware test: A new test approach towards dppb level," in *2019 IEEE International Test Conference (ITC)*, 2019, pp. 1–10.
[19] ——, "Structured test development approach for computation-in-memory architectures," in *2022 IEEE International Test Conference in Asia (ITC-Asia)*, 2022, pp. 61–66.
[20] ——, "Testing scouting logic-based computation-in-memory architectures," in *ETS*, 2020.
[21] A. Singh *et al.*, "Accelerating rram testing with low-cost computation-in-memory based dft," in *ITC*, 2022.
[22] J.-H. Lee *et al.*, "Exploring cycle-to-cycle and device-to-device variation tolerance in MLC storage-based neural network training," *TED*, 2019.
[23] Y. Jeong *et al.*, "Parasitic effect analysis in memristor-array-based neuromorphic systems," *Nanotech*, 2018.
[24] P. Chen *et al.*, "Technological Benchmark of Analog Synaptic Devices for Neuroinspired Architectures," *IDT*, 2019.
[25] M. Fieback *et al.*, "Defects, fault modeling, and test development framework for rrams," *JETC*, 2022.
[26] S. Ambrogio *et al.*, "Reducing the impact of phase-change memory conductance drift on the inference of large-scale hardware neural networks," in *IEDM*, 2019.
[27] W. Shim *et al.*, "Investigation of read disturb and bipolar read scheme on multilevel rram-based deep learning inference engine," *TED*, 2020.
[28] A. Singh *et al.*, "Cim-based robust logic accelerator using 28 nm stt-mram characterization chip tape-out," in *AICAS*, 2022.
[29] M. Cheng *et al.*, "Time: A training-in-memory architecture for memristor-based deep neural networks," in *DAC*, 2017.
[30] A. Singh *et al.*, "Referencing-in-array scheme for rram-based cim architecture," in *DATE*, 2022.
[31] S. Diware *et al.*, "Unbalanced bit-slicing scheme for accurate memristor-based neural network architecture," in *AICAS*, 2021.
[32] P. C. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," in *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, 1996, pp. 104–113.
[33] Y. Kim *et al.*, "Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors," in *ACM/IEEE 41st International Symposium on Computer Architecture, ISCA 2014, Minneapolis, MN, USA, June 14-18, 2014*. IEEE Computer Society, 2014, pp. 361–372.
[34] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," *SIAM Rev.*, vol. 41, no. 2, pp. 303–332, 1999.
[35] P. C. Kocher *et al.*, "Differential Power Analysis," in *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, ser. Lecture Notes in Computer Science, M. J. Wiener, Ed., vol. 1666. Springer, 1999, pp. 388–397.
[36] K. Gandolfi *et al.*, "Electromagnetic analysis: Concrete results," in *Cryptographic Hardware and Embedded Systems - CHES 2001, Third International Workshop, Paris, France, May 14-16, 2001, Proceedings*, ser. Lecture Notes in Computer Science, Ç. K. Koç *et al.*, Eds., vol. 2162. Springer, 2001, pp. 251–261.
[37] E. Biham *et al.*, "Differential Fault Analysis of Secret Key Cryptosystems," in *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, 1997, pp. 513–525.
[38] F. Staudigl *et al.*, "Fault Injection in Native Logic-in-Memory Computation on Neuromorphic Hardware," 2023.
[39] Z. Wang *et al.*, "Side-Channel Attack Analysis on In-Memory Computing Architectures," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–13, 2023.
[40] D. Lee *et al.*, "Keystone: An open framework for architecting trusted execution environments," in *Proceedings of the Fifteenth European Conference on Computer Systems*, 2020, pp. 1–16.
[41] D. Kohlbrenner *et al.*, "Building open trusted execution environments," *IEEE Security & Privacy*, vol. 18, no. 5, pp. 47–56, 2020.
[42] A. Giron *et al.*, "Post-quantum hybrid key exchange: a systematic mapping study," *Journal of Cryptographic Engineering*, vol. 13, pp. 1–18, 04 2022.
[43] T. Pornin, "New efficient, constant-time implementations of falcon," Cryptology ePrint Archive, Paper 2019/893, 2019, https://eprint.iacr.org/2019/893.
[44] S. Kundu *et al.*, "Higher-order masked saber," in *Security and Cryptography for Networks: 13th International Conference, SCN 2022, Amalfi (SA), Italy, September 12–14, 2022, Proceedings*. Berlin, Heidelberg: Springer-Verlag, 2022, p. 93–116.
[45] J. Wichelmann *et al.*, "Microwalk: A framework for finding side channels in binaries," in *Proceedings of the 34th Annual Computer Security Applications Conference, ACSAC 2018, San Juan, PR, USA, December 03-07, 2018*. ACM, 2018, pp. 161–173.
[46] S. Chari *et al.*, "Towards sound approaches to counteract power-analysis attacks," in *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, ser. Lecture Notes in Computer Science, M. J. Wiener, Ed., vol. 1666. Springer, 1999, pp. 398–412.
[47] H. Bar-El *et al.*, "The Sorcerer's Apprentice Guide to Fault Attacks," *Proc. IEEE*, vol. 94, no. 2, pp. 370–382, 2006.
[48] Y. Ishai *et al.*, "Private Circuits: Securing Hardware against Probing Attacks," in *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, ser. Lecture Notes in Computer Science, D. Boneh, Ed., vol. 2729. Springer, 2003, pp. 463–481.
[49] J. Richter-Brockmann *et al.*, "Revisiting fault adversary models - hardware faults in theory and practice," *IEEE Trans. Computers*, vol. 72, no. 2, pp. 572–585, 2023.
[50] F. McKeen *et al.*, "Innovative instructions and software model for isolated execution." *Hasp@ isca*, vol. 10, no. 1, 2013.
[51] ARM Ltd., "ARM security technology building a secure system using TrustZone technology," *Whitepaper*, 2016.
[52] A. Nilsson *et al.*, "A survey of published attacks on Intel SGX," *arXiv preprint arXiv:2006.13598*, 2020.