Design and analysis of an adaptive radiation resilient RRAM subsystem for processing systems in satellites

Daniel Reiser[1] · **Junchao Chen[2]** · **Johannes Knödtel[1]** · **Andrea Baroni[2]** · **Miloš Krstić[2,3]** · **Marc Reichenbach[1]**

Abstract

Among the numerous benefits that novel RRAM devices offer over conventional memory technologies is an inherent resilience to the effects of radiation. Hence, they appear suitable for use as a memory subsystem in a computer architecture for satellites. In addition to memory devices resistant to radiation, the concept of applying protective measures dynamically promises a system with low susceptibility to errors during radiation events, while also ensuring efficient performance in the absence of radiation events. This paper presents the first RRAM-based memory subsystem for satellites with a dynamic response to radiation events. We integrate this subsystem into a computing platform that employs the same dynamic principles for its processing system and implements modules for timely detection and even prediction of radiation events. To determine which protection mechanism is optimal, we examine various approaches and simulate the probability of errors in memory. Additionally, we are studying the impact on the overall system by investigating different software algorithms and their radiation robustness requirements using a fault injection simulation. Finally, we propose a potential implementation of the dynamic RRAM-based memory subsystem that includes different levels of protection and can be used for real applications in satellites.

✉ Daniel Reiser
daniel.reiser@uni-rostock.de

Junchao Chen
chen@ihp-microelectronics.com

Johannes Knödtel
johannes.knoedtel@uni-rostock.de

Andrea Baroni
baroni@ihp-microelectronics.com

Miloš Krstić
krstic@ihp-microelectronics.com

Marc Reichenbach
marc.reichenbach@uni-rostock.de

[1] Institute of Applied Microelectronics and Computer Engineering, University of Rostock, Rostock, Germany

[2] IHP, Leibniz Institut für innovative Mikroelektronik, Frankfurt Oder, Germany

[3] Institute of Computer Science, University of Potsdam, Potsdam, Germany

Springer

1 Introduction

Recent technological advancements as well as cost reductions in satellite and launch vehicle development have expanded access to space. Notably, by the end of 2022, the aerospace domain witnessed a 38.4% increase in the number of orbiting satellites from the previous year, reaching a total of 6718 [1]. Small satellites, in particular, now support diverse research and commercial ventures, including imaging, monitoring, communications, and navigation [2]. Applications range from traditional algorithms for processing images, navigation or data routing to applications based on artificial intelligence. Since remote connections might be unreliable or bandwidth-limited, and data transmission between a satellite and earth can introduce considerable latency, on-board, real-time data processing is often a necessity. Therefore, applications with elaborate calculations in space also need fast and robust processing platforms.

However, the space radiation environment is both dynamic and complex [3]. It poses challenges to microelectronic circuits due to extreme thermal variations and high-energy radiation. These challenges, along with the scaling of Integrated Circuits, make processing systems in satellites vulnerable to high-energy particles such as protons and heavy ions. In particular, the occurrence of solar particle events (SPEs) can lead to rapid spikes in radiation levels by several orders of magnitude in a short period of time. This exposure can lead to destructive and non-destructive anomalies in electronic circuits, causing data corruption and even system failures. In addition, the components are difficult to access and no manual repairs can be carried out in space. In light of these challenges, there is an urgent need for reliable electronic systems to mitigate radiation-induced effects.

There are several approaches for creating a radiation resilient processing system for space, reaching from systems based on commercial off-the-shelf (COTS) components, to specific computer architecture designs for use in harsh environments. Examples include fault-tolerant processing systems based on FPGAs, systems based on RISC-V, or custom CPUs [4–6].

However, one central and important aspect of such systems is often neglected and has not been further optimized in recent years, namely the memory subsystem. Most processing systems to this day utilize memory subsystems consisting of a combination of local SRAM and external DRAM memory. Circuit designers commonly employ static fault mitigation methods, such as error detection units, error correction codes (ECC), and redundant information storage, based on worst-case radiation scenarios to harden the memory.

However, on one hand these hardening measures are unnecessary under many conditions, leading to undue power consumption and performance degradation. On the other hand such classical electron-based memories possess a far inferior inherent radiation resilience, than novel memory technologies such as resistive RAMs (RRAMs). RRAMs have proven to be inherently radiation tolerant, by using ions instead of electrons to store information [7].

In addition to their inherent resilience against radiation, RRAMs posses many advantages over electron-based memories such as SRAM or DRAM. They are highly power efficient, ensuring extended runtimes in energy-constrained environments. They offer space efficiency, which reduces the system's physical size. In addition, RRAM cells provide non-volatile data storage that prevents data loss during power outages in a satellite. Furthermore, they can store multiple bits of information in a single cell, resulting in high storage density in small

Fig. 1 Block diagram of the overall evaluation methodology for our system
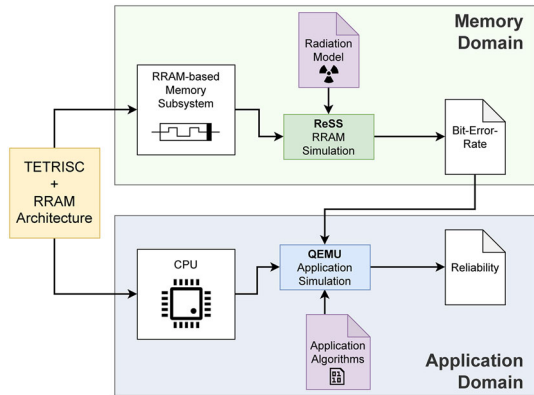
memory footprints. The combination of these features makes RRAMs an excellent choice for use as an embedded memory subsystem in a processing platform for satellites.

Contrary to static protection measures for the memory subsystem, a more efficacious approach entails activating circuit protection only when required and thereby ensuring an optimal balance between power consumption, performance, and reliability [8]. In space applications, real-time detection and prediction of multiple sources of failure, such as radiation, is essential in order to cope with the complex and changing space environment and to achieve the deployment of optimal reinforcement methods under various conditions [9].

Therefore we introduce in this article for the first time a RRAM-based memory subsystem design for a satellite processing system that can dynamically respond to radiation events. This system provides a reliable and energy-efficient solution for radiation hardness, and it can seamlessly switch between reliability, performance and low-power modes. To detect and forecast radiation events, such as those emanating from SPEs, the RRAM-based memory subsystem is integrated into an established satellite system-on-chip (SoC) platform, TETRISC [9].

This platform encompasses an in-built radiation monitor network unit, enabling real-time detection and prediction of the radiation environment, including specific events like SPEs. Furthermore, based on the detected or predicted radiation conditions, this processing cores can dynamically reconfigure to the optimal operation mode, ensuring a balanced trade-off between performance, power consumption, and reliability. We now apply the same principles used in this platform to the RRAM-based memory subsystem, while at the same time adapting them for the unique physical attributes of RRAM devices.

In the following we present to our knowledge the first SoC platform design for space with a RRAM-based memory subsystem. Furthermore we introduce the tools, developed to evaluate the performance of the system under radiation on memory level and on application level. An overview over this evaluation concept is shown in Fig. 1. The evaluation of the TETRISC+RRAM architecture is split up into the simulation of the RRAM-based memory subsystem and the simulation of application algorithms on the system under radiation. The memory simulation gives performance metrics for the Bit-Error-Rate of the memory under different radiation levels in different operation modes. The application simulation shows if and how well different algorithms will work on the system during a radiation event. Ultimately, we provide an overview about the capabilities of the new adaptive memory subsystem design and elaborate on possible strategies for the most efficient use of the platform. With this, we demonstrate that combining RRAM memory with a meticulously designed SoC,
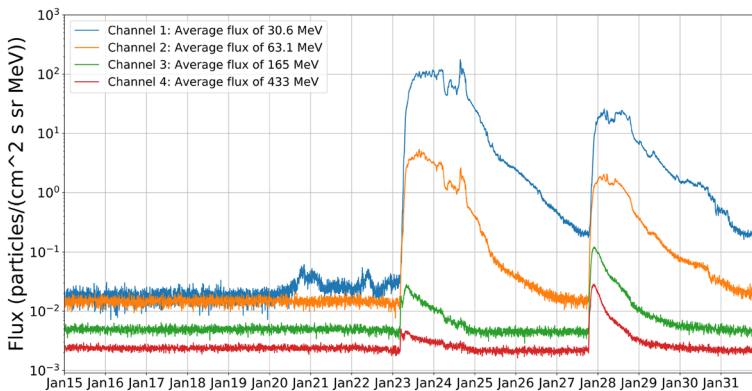
Fig. 2 Example of proton flux variations at different energy levels during the solar event in January 2012 obtained from the GOES-13 satellite

paves the way for an energy-efficient, radiation-hardened memory system tailored for space applications.

2 Related work

2.1 Space radiation

Space-borne applications' design and adjustment are heavily influenced by the space radiation environment, accounting for approximately 20% of satellite anomalies [3]. Thus, mitigating radiation-induced effects in space electronics is paramount. Space radiation primarily emanates from two sources: planetary magnetospheres, exemplified by Earth's Van Allen belt, and transient deep-space radiation. Planetary magnetic fields trap charged particles such as protons, electrons, and heavy ions. Deep space radiation predominantly consists of Galactic Cosmic Rays (GCRs) and SPEs, encompassing protons and heavy ions from all periodic table elements [10]. Explosive events, like supernovae, give rise to GCRs outside the solar system. In contrast, SPEs, often instigated by large-scale magnetic eruptions due to swift Coronal Mass Ejections (CMEs) and associated solar flares, markedly transform the radiation landscape, inducing a surge in energetic particles.

SPEs, upon occurrence, rapidly dominate the space radiation environment. Such events cause an abrupt rise in energetic protons, ions, and electrons within interplanetary space. The flux during an SPE can peak in mere minutes or hours and then wane over subsequent hours or days. This peak flux can surpass background space radiation levels by two to five orders of magnitude. Even with Earth's protective magnetic field and atmosphere, SPEs profoundly influence the terrestrial radiation environment, potentially boosting ground-based cosmic ray levels by up to 5000% during especially intense events [10]. In Fig. 2, data derived from GOES-13's EPEAD and HEPAD particle detectors demonstrate the behavior of space proton flux across various energy ranges from 15 January to 31 January 2012, a period marked by significant solar activity [11]. Four channels, selected from both low and high-energy ranges, were used to illustrate the rapid enhancement in proton flux during solar particle events. The National Oceanic and Atmospheric Administration (NOAA) characterizes the onset and termination of an SPE using specific proton flux data points with energies $\geq$ 10 MeV [12].

Since 1976, NOAA recorded 267 SPEs impacting Earth, with an annual frequency varying between 0 and 30. Some events were severe enough to harm electronic systems.

High-energy particles pose a significant threat to the dependability of advanced ICs in two ways: (1) transient effects like bit flips in memory cells, and (2) enduring effects such as threshold voltage drifts and mobility deterioration. Transient effects encompass soft errors in Single Event Effects (SEEs) caused by individual high-energy particles, leading to either soft or hard faults in ICs. On the other hand, prolonged radiation consequences arise from Total Ionizing Dose (TID) and displacement damage, entailing physical processes like bias temperature instability causing lasting electrical degradation. Data from historical space expeditions [13, 14] highlight that fault rates could increase manifold during SPE periods relative to non-SPE conditions. Given the amplified fault likelihood during high-particle-activity episodes, continuous monitoring of space radiation, forecasting SPEs, and deploying appropriate electronic protection are imperative.

Based on data provided by NOAA [12], a total of 36 solar events influenced Earth's environment during solar cycle 24 (2008–2019). A meticulous analysis of the corresponding space proton and heavy-ion fluxes reveals that these solar events cumulatively lasted approximately 2500 h, translating to an average occurrence of roughly 2.5% of an annual timeframe, and the majority of the time being a non-SPE phase. This fact shows that optimizing an electrical system for dynamic resilience can bring great benefits if the resources freed up as a result can be used differently 97.5% of the time. A more detailed analysis of the historical solar events can be found in our previous work [15].

2.2 RRAM-based systems for space

RRAM devices, with their numerous advantages over commonly utilized memory devices like SRAM or DRAM, have been investigated for use in various application scenarios. Due to their compact size, energy efficiency and high storage density, their suitability for space applications has also been examined. In addition to terrestrial applications, the impact of radiation on these devices has been widely studied [7, 16–18]. Furthermore, experiments with RRAM devices have been conducted in space to study real world performance and reliability in extreme environments [19]. Also the design of the layers of the RRAM devices has been optimized specifically for radiation hardness [20]. It has been shown, that RRAM devices possess inherent radiation resilience. However, errors in form of changes in conductance can be introduced. Mainly because of the susceptibility of the access transistor, that is used to program and read such devices.

Therefore, various strategies have been employed to mitigate radiation effects, including the development of analog circuitry for radiation-hardened RRAM latches [21] or analog readout circuitry with error detection and correction capabilities [22]. Alternatively, in-memory error correction codes have also been implemented to mitigate radiation-induced faults, ensuring reliable RRAM device operation even in challenging radiation environments, such as space [23]. However, all these strategies share the common feature that their radiation hardness remains active throughout the entire operation time. To our knowledge, no approach can dynamically respond to radiation events that are currently occurring or predicted. This fact, combined with the rarity of dominant SEUs during runtime, suggests that implementing an adaptive solution could lead to potential energy savings or increased performance.

3 TETRISC + RRAM platform

The TETRISC (TETra Core System based on RISC-V) SoC, a quad-core system based on the RISC-V architecture and constructed on the PULPissimo platform, presents an adaptive approach to ensuring system reliability in harsh environments such as space [9]. Dynamic high-reliable adaptive systems should autonomously adjust their behavior when they do not meet requirements and ensure reliable service provision, even in the face of anomalies or faults. Presently, FPGAs are the primary choice for dynamic designs, as highlighted by studies such as Jacobs et al. [24] and R. Glein et al. [25]. On the other hand, while ASIC-based designs offer less flexibility, they possess distinct configurations as demonstrated in [26] and [27]. However, unlike the proposed TETRISC SoC platform, to the best of our knowledge, no other solution can adaptively protect the system by considering both real-time detected and predicted radiation conditions as well as the system state.

Designed for dynamic adaptability, the TETRISC SoC adjusts its reliability in real-time, balancing reliability, power consumption, and performance while monitoring multiple fault sources. It integrates various reliability monitors, including those for Single Event Upsets (SEU), core aging, and temperature. The SoC's architecture enables it to detect and forecast environmental conditions, allowing its four cores to operate across diverse performance and fault tolerance modes based on real-time data.

The SEU Monitor [28] within the TETRISC SoC is a RAM-based, non-standalone embedded system designed to monitor in-flight radiation-induced SEU rates in real-time, ensuring prompt detection of radiation levels and safeguarding radiation-sensitive circuits. It employs standard on-chip SRAM memory as a particle detector. Utilizing the scrubbing approach, Error Detection and Correction (EDAC) code, and the over-counting detection register file with a dedicated detection flow, the proposed monitor ensures accurate counting of all upsets occurring in the target SRAM and distinguishes between error types (i.e., single-bit upsets, multiple-bit upsets, and permanent faults) in each memory word. A more detailed explanation can be found in our previous research [15, 28, 29]. Complementing this, the SEU monitor can also feature a predictor tailored for on-chip hourly space radiation environment predictions, leveraging real-time SEU data and a pre-trained machine learning model [15]. The methodology divides into two phases: offline data analysis and online predictive monitoring. The offline phase taps into historical space radiation datasets, mainly from ACE-SIS and GOES databases (for heavy-ion and proton data), to develop a machine learning prediction model. After meticulous evaluations, a linear regression approach stands out for its efficiency and precision. Besides that, the online phase employs the trained models on real-time SEU data, focusing on predicting the SEU rate and the associated radiation conditions. This solar event predictor, by synthesizing historical data insights with real-time metrics, bolsters the adaptive hardening mechanisms of space-borne systems against solar events.

The HiRel Framework Controller (HFC) serves as the main adaptive control subsystem of the TETRISC SoC, allowing for hardware-based reconfigurability and fault tolerance. The HFC enables diverse operational modes based on core-level N-Module Redundancy (NMR) and clock-gating techniques. It can transition from a 2MR to a 4MR system with any combination of active processors, creating an NMR on-demand system. A key feature of the HFC is the programmable majority voter [30], which enhances the system's adaptability and reliability. With its programmable design, the voter can adjust its voting inputs based on different reliability needs. Using the Input State Descriptor (ISD), the voter quickly identifies and corrects system faults. It also has a self-checking feature, which improves the system's overall reliability. Additionally, the HFC manages a real-time monitoring system, selecting
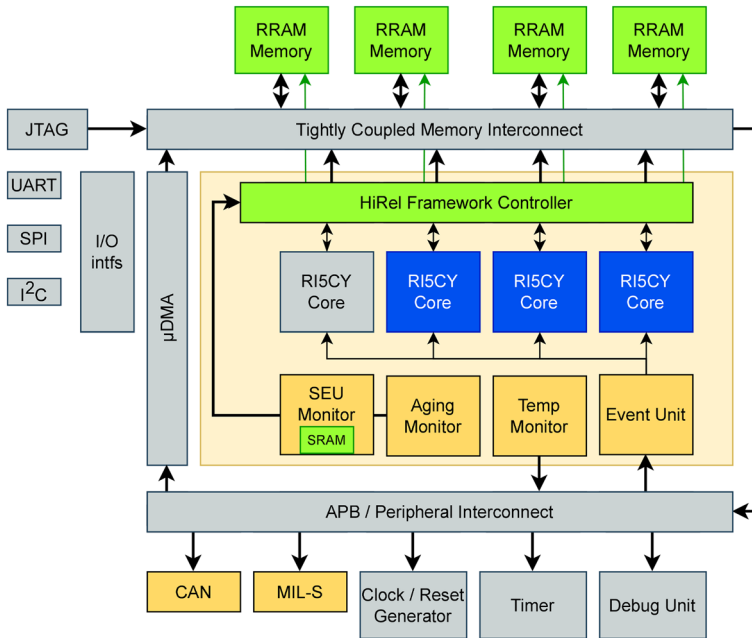
Fig. 3 Block diagram of the TETRISC platform including the RRAM memory subsystem and modified HiRelFramework Controller

the best operational mode. It also activates radiation-hardening methods based on current data and predictions from the monitoring system. Overall, the HFC is crucial for mixed-criticality systems, ensuring the SoC's suitability for challenging environments.

Previously, the TETRISC platform's focus was on providing a runtime configurable, resilient processing system, that can switch between fault-tolerance, high-performance and power-saving modes, depending on the environmental conditions. In this work, we shift the scope to the design of a configurable and resilient memory subsystem for the platform, that extends upon the same concept. To achieve this we replace the platform's SRAM with RRAM memory and tweak the system design to accommodate for the features this new memory offers.

Figure 3 depicts a block diagram of the new TETRISC + RRAM platform, where components in grey are the PULPissimo platform's base components. Blue and yellow components were integrated by the TETRISC design and green blocks indicate the components that underwent modification or are newly introduced in this work. The most significant modification done is the replacement of the original SRAM memory banks with 40 kB each with RRAM memory banks of the same size. The memory specifications of four 40 kB banks are maintained according to the original TETRISC design for better comparison. However, the platform allows for scaling up the RRAM size, which enhances its application spectrum. Four such RRAM memory banks are incorporated. For all four banks a power-down signal is introduced. In this way, the voltage supply of the RRAM memory banks that are not used can be switched off completely to save energy (power-gating). Since the information is stored inside the RRAM cells in a non-volatile manner, all data is immediately available when the power supply is switched back on again. Therefore no energy overhead for reloading the data is introduced compared to SRAM. At the same time 2 bits of information can be reliably
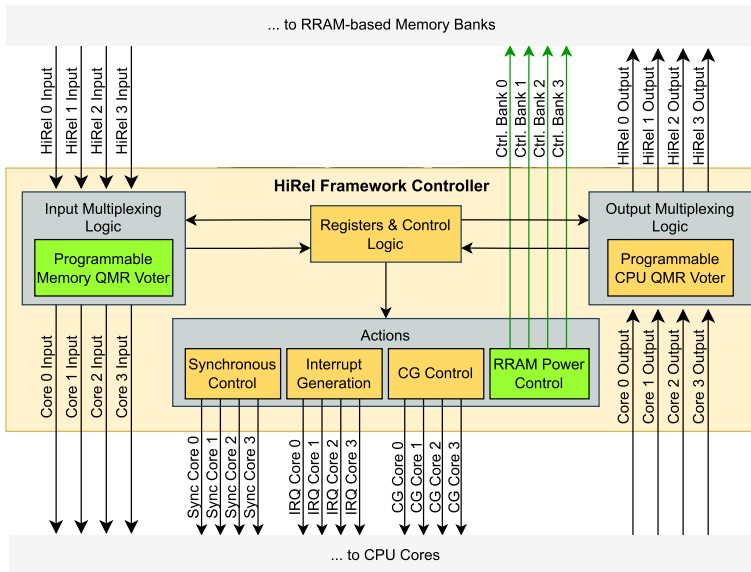
Fig. 4 Block diagram of the TETRISC HiRel Framework Controller including the modifications to control the RRAM-based memory banks

stored in a single RRAM cell, meaning a reduction of up to $12\times$ the amount of transistors compared to SRAM memory. It is worth noting that a small amount of SRAM memory is integrated into the SEU monitor, which allows the current prediction methodology based on solar event detection in SRAM to continue to work.

The modifications made to the HFC are shown in more detail in Fig. 4. To efficiently respond to anticipated solar events and dynamically switch the CPU between fault-tolerant, high-performance and power-saving mode, this subsystem includes various control logic, as well as a programmable majority voter for the CPU content. In fault-tolerant multiple CPU cores calculate the same result and the majority voter decides which result will be transferred to memory. A new programmable majority voter is introduced for memory content, so it too can be switched to a fault-tolerant storage mode. Additionally, an RRAM Power Control unit is integrated, which can issue power-down signals to the individual RRAM memory banks. In case of an upcoming solar event, the fault-tolerance mode can be activated by switching on memory banks for redundant data storage and the majority voter at the same time. For normal operation, either high-performance mode using all memory banks in parallel or power-saving mode using only one memory bank and powering off the other can be selected.

4 Methodology

4.1 RRAM simulation framework

To simulate functional as well as non-functional properties of RRAM devices in a limited time, we developed a SystemC-based RRAM simulation framework, called ReSS. An overview of it is depicted in Fig. 5. This framework mainly consists of two parts: the "in-memory" block, where RRAM arrays are simulated. The conductance behaviour is described
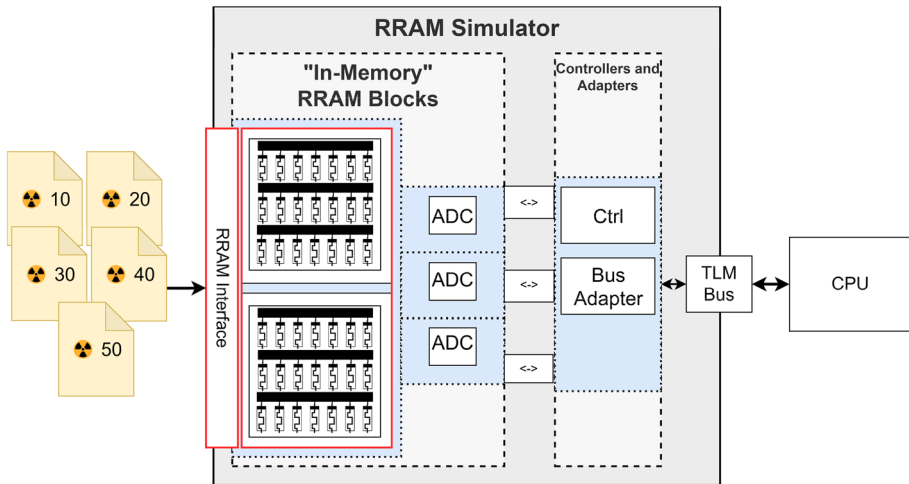
Fig. 5 RRAM Memory simulator with interface for measured RRAM effects in SystemC

by a statistical model implemented in C++. A unified interface is used to load statistical properties of RRAM cells extracted from real RRAM measurements. Using a Monte-Carlo approach different variances in different environmental conditions can be simulated with low performance overhead. The other part consists of the simulation of memory control units and bus adapters to provide access to the RRAM memory subsystem from external devices. Through a SystemC TLM bus data can be written and read to and from the simulated RRAM arrays. In this work the evaluation of the radiation induced errors is split into a simulation in the memory domain and a simulation in the application domain. The ReSS framework is used for the simulation of radiation effects on memory itself and possible mitigation strategies. Memory access from the TETRISC platform to the RRAM memory banks through the TLM bus is evaluated with and without the influence of radiation. As a resulting metric the Bit-Error-Rate (BER) and power draw of the system for different radiation intensities and mitigation strategies are given. The implementation details of the SystemC model have no influence on these results.

The behaviour of the RRAM memory banks under radiation is simulated by using a radiation model loaded into the ReSS framework via the RRAM interface. The model used is an abstract model, we derived from measurements of actual RRAM devices produced by IHP [31]. The measurements where taken after the programming algorithm was completed and multiple thousands of devices have been used to achieve a statistically correct extrapolation. With the help of these measurements the conductance distribution of the four different states, stored in each device are characterized. To describe the conductance behaviour, normal distributions are fitted to the measurements. The states used for the IHP RRAM devices are called L1, L2, L3 and L4 and have a target conductance of $50\,\mu S$, $100\,\mu S$, $150\,\mu S$ and $200\,\mu S$ respectively. Currently, these devices have not been measured under radiation. However, their behaviour under radiation can be extrapolated using the measurements under normal environmental conditions as baseline and adding characterizations of similar devices under radiation to the model. An abstract model was derived based on studies on Single-Event (SEE) and Multi-Event (MEE) Upsets as well as Total ionizing dose effects (TID) on similar RRAM devices [16–18]. Radiating the RRAM device will mostly have the same effect as unintentionally programming a device. The induced current will increase the conductive

Fig. 6 Model of the radiation effect on different RRAM conductance states for increasing radiation intensity
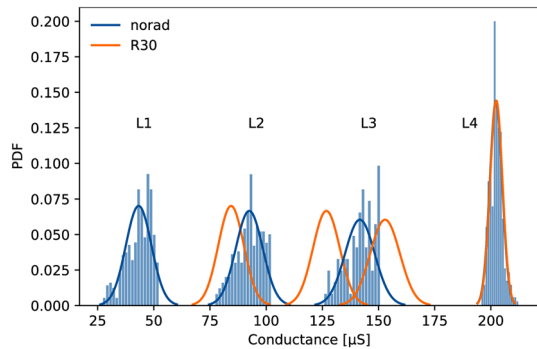
filament of the devices depending on their initial state. Low-conductive (or high-resistive) states will be more severely affected by this effect than high-conductive (or low-resistive) states. The conductivity of initially low-conductive states will increase towards more high-conductive states when under radiation. The strength of the conductance shift depends on the number and energy of the radiation particles. A reprogramming or writing to the devices will revert the radiation effects. Because of missing measurements, we cannot directly correlate the radiation intensity and energy to the shift in conductance. Therefore, to simulate different intensities of radiation, we defined six operating conditions for our radiation model. Two conditions are depicted in Fig. 6. The non-radiated operation ('norad') depicts a probability density function plot of the normal distributions for the distinguishable states as well as a histogram of the actual measured conductance values from the IHP RRAM devices. Radiation conditions R10, R20, R30, R40, and R50 are then defined, in which the mean of the conductance distributions of each state is shifted towards higher conductance respectively. The states with higher conductance are exponentially less effected by the same radiation intensity [16]. Therefore, the shift towards higher conductance is modelled as exponentially decreasing for the three higher conductive states as well as higher radiation intensity. This can be seen at the example of R30.

The coefficients for our exponential model where found by interpolation of radiation measurement results of RRAM devices from literature [16–18] and applying the shift in resistance (or conductance) to the measurements of the unradiated IHP devices. The resulting model parameters are displayed in Table 1.

To compare different mitigation strategies for a resilient RRAM memory subsystem, a simulation model for the power of the memory system is utilized. This model describes the energy needed for reading and writing/programming a RRAM cell, along with the static power dissipation of the system. The model includes the 1T1R cells itself, as well as the analog circuitry to access the devices. The circuit design and power model are based on the work by [32]. This model is computed using a script that takes into account the number of memory accesses and the clock frequency exported by our simulator.

4.2 Application simulation framework

As demonstrated in the previous chapter, the TETRISC SoC employs RISC-V processors for its computational tasks. To assess the functional consequences of radiation-induced errors in the RRAM memory subsystem on the target platform, we conduct a fault simulation in the application domain. This simulation is based on an instruction set emulator for the RISC-

Table 1 Parameters of our radiation model of the form: $Conductance = a \cdot b^n$

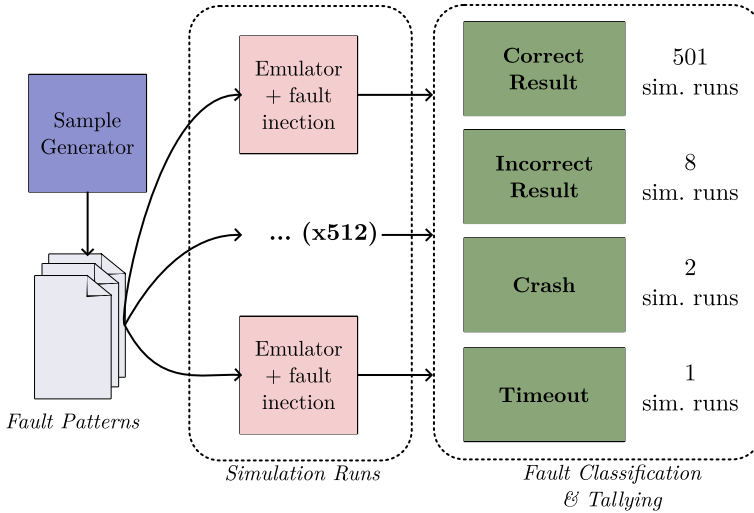| Conductance level	a	b
L1	43.2	1.022565
L2	92.47	1.010592
L3	141.7	1.002535
L4	202.3	1.000009
Radiation intensity		n
R10		1
R20		2
R30		3
R40		4
R50		5

Fig. 7 Block diagram of the fault injection methodology

V architecture and utilizes the widely-adopted QEMU platform. This methodology offers several advantages compared to more conventional approaches, such as netlist-based fault simulation or execution of algorithms on the real hardware. First and foremost, it exhibits exceptional efficiency, enabling us to conduct in-depth analyses involving a multitude of programs and diverse environments. For the results presented in Sect. 6, we executed over 2 million simulations. Secondly, this method remains agnostic to the specific microarchitecture of the processor since it exclusively focuses on the functional aspects of memory faults, thus ensuring broad applicability and relevance.

Figure 7 shows a block diagram outlining the methodology. The core concept involves conducting multiple simulations with injected faults into load instructions, with the fault injection rate determined by the results of the RRAM-based memory subsystem simulation detailed in Sect. 4.1. The outcome of the application simulation yields the probability of successful program execution under specific radiation conditions.
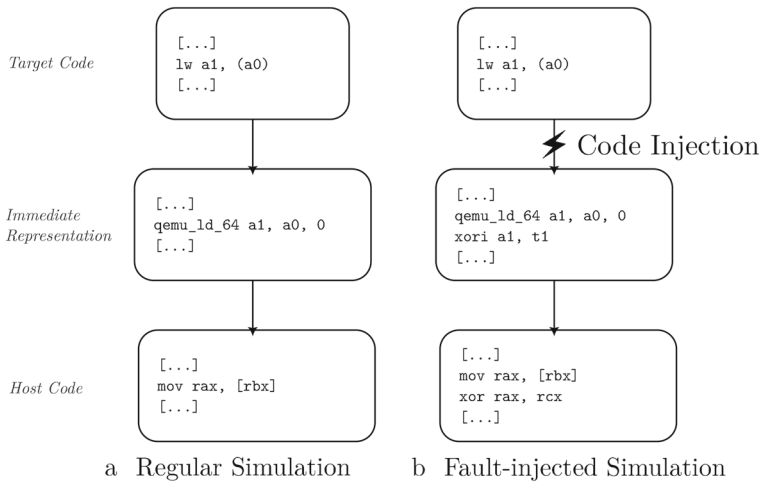
Fig. 8 QEMU fault injection process

The QEMU emulator is a free and open-source emulator for a variety of host and target platforms. It is known for its performance and feature set, which is achieved through the use of a Just-In-Time (JIT) compiler as the main emulation facility. When running any code in QEMU, it first gets translated into an immediate representation (IR), which then gets translated to host code. This process can be seen in Fig. 8a. The `lw` (Load Word) instruction in the target code gets translated to a `qemu_ld_64` (Load 64-Bit Integer) instruction in the immediate representation, which in turn gets translated to a `mov` instruction. This instruction simulates the virtual memory access by loading the value from memory to a register. We modify this simulation process to evaluate the influence of radiation induced errors in the memory. By injecting code into the immediate representation of the load instructions, bit flips are introduced to the result of the memory operation, as shown in Fig. 8b. Here an additional instruction (`xori`, XOR with immediate) is added during the Target Code to Immediate Representation translation. This instruction alters the register containing the result of the load instruction. In detail injected code simulates radiation induced memory errors in the following way:

1. When translating a load instruction to the target code, additional IR instructions are injected, directly after the regular IR generated for the instruction.
2. From a large global pool of samples a 64-bit value is drawn, which will be called "fault mask". This pool consists of 64-bit values provided to the simulation on start-up.
3. This fault mask is XOR-ed with the actual result of the load instruction. In effect this leads to a bit-flip in the loaded data of each bit where the sample was 1.
4. The index used to draw a fault mask from the pool is incremented.

With this method, the number, time and location of faults can be precisely controlled. At the same time the simulation speed is not increased significantly. It needs to be noted, that our simulation currently only considers data memory and does not cover instruction memory, registers or internal states. As preparation for the simulation, a pool of fault masks needs to be created matching the BER of the simulated RRAM memory subsystem configuration. In order to give insights into the different error modes and their likelihood, multiple pools for a given BER are generated. For the simulations presented we classified each run of

the emulator into "correct result", "incorrect result", "crash" and "timeout". Crashes and timeouts are detected by the emulation environment, while the correctness of the output is determined by the application, after reverting back to a simulation without fault injection. For example, when evaluating a matrix multiplication, the resulting matrix will be compared to a reference after execution. If any entry differs, the simulation run is classified as "incorrect result". A crash is any situation where the operating system would prematurely terminate the application, e.g. when it accesses memory at a currently unmapped location. In the matrix multiplication example, the index of a loop might get stored in memory and the load to this location gets corrupted leading to a memory access outside of the array backing the matrix. The timeout category refers to executions, where the program does not terminate anymore within an reasonable time in certain error conditions. This problem can occur when for example a variable controlling the execution flow gets corrupted and the program runs into an endless loop.

5 Evaluation

It has already been shown that RRAM devices can be considered inherently resilient regarding radiation [17]. However, for high radiation intensity errors might still occur in our RRAM memory banks, especially when up to four states are stored in one single RRAM cell.

Using our SystemC based ReSS framework, we simulate the memory subsystem with its four RRAM memory banks with and without the influence of radiation. As described in Sect. 3 four RRAM-based memory banks with 40 kBytes of storage each using 2 bits (or four states) per cell are simulated. Generating a random value based on the statistical radiation model determines the conductance of each RRAM cell during read operations. If the output conductance of a simulated cell is closer to the target conductance of a different state, than the one originally stored, a misclassification occurs, and bit flips are introduced into the resulting data. This behavior is imposed by the design of the ADC used to read the data from the cells. In this memory domain simulation we evaluate the rate with which a bit flip happened compared to the overall read bits. This metric is known as the BER.

The BER simulation will be run for the six different radiation levels, defined by our radiation model (see Sect. 4.1). We will iteratively evaluate different system configurations starting with the unprotected or performance mode of the memory subsystem. Then we will introduce different mitigation strategies that can be applied in the fault-tolerant mode of the system. We will check how good they perform and what improvements can be made to reduce the BER. For comparison we will analyse the estimated power draw of the system with the mitigation strategies applied an compare it to the power draw of the unprotected system.

The baseline results of the unprotected system are listed in Table 2(a). For an unprotected RRAM memory subsystem, the BER is almost 0% for the R10 and about 2% for the R20 intensity configuration. For R30 the BER climbs to 27% and for R40 and R50 to 36% and 37% respectively. The estimated power draw for this basic configuration is about 208 mW.

5.1 TMR and QMR

The TETRISC platform already provides error mitigation strategies for the CPU side in form of Triple-Modular-Redundancy (TMR) and Quad-Modular-Redundancy (QMR) using a programmable majority voter [9]. We introduced the same voter also for the memory subsystem to protect the RRAM content from radiation induced errors. In the TMR or QMR

Table 2 BER and average power draw results for the RRAM-based memory subsystem under radiation

		BER						
	Power (mW)	Unradiated (%)	R10 (%)	R20 (%)	R30 (%)	R40 (%)	R50 (%)
(a)	Unprotected	208	0	0.03	2.1	27	36	37%
(b)	TMR	311	0	0	0.4	29	36	37
(c)	QMR	363	0	0	0.6	29	37	38
(d)	extended TMR	311	0	0	0.08	15	24	25
(e)	extended QMR	363	0	0	0.08	15	24	25
(f)	1-Bit Cells	415	0	0	0	0	0	0
(g)	extended 1-Bit Cells	260	0	0	0	0	0	0

The average power draw includes power-saving and fault-tolerance modes. The latter is only used 2.5% of the time
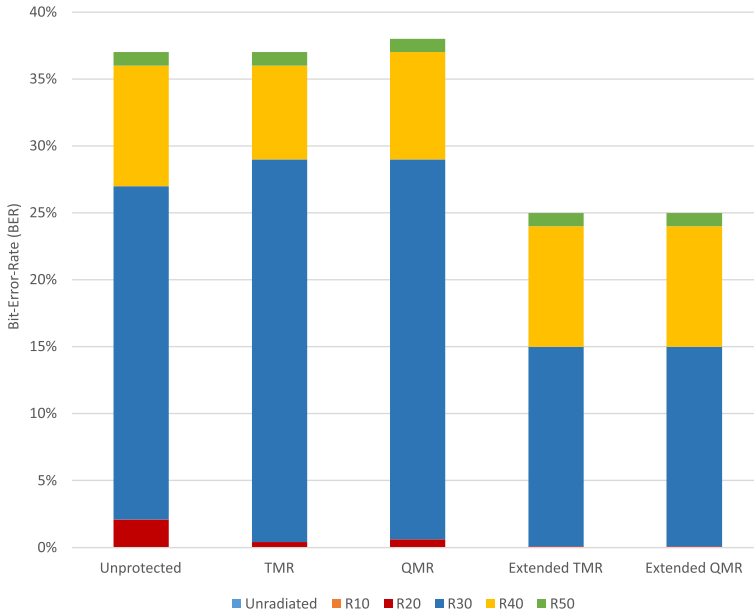
Fig. 9 BERs for different radiation mitigation strategies of the RRAM memory subsystem

operation mode a write command with the same data is issued to three or four memory banks in parallel and the programmable majority voter will be used during a read operation to determine which data is the most plausible. Since all RRAM devices used in parallel are exposed to the same radiation influence and the information is encoded with the same state in all of them, the probability of an incorrect classification is equally high in all devices. Statistically, the probability of a correct result for a majority voting with three input values is therefore just as high as with four input values. Nevertheless, we examine the QMR strategy because we expect that changing the mapping of data to RRAM states, as we will do later, could improve the QMR strategy.

We simulated the BER and power draw for the memory subsystem using TMR and QMR for protection. The results are listed in Table 2(b) and (c). A visual representation can be found in in Fig. 9. It can be seen, that for low radiation intensity up to R20, TMR and QMR achieve a decrease in BER to < 1%. As expected, the error correction capability is equally good for both strategies. For higher intensity TMR and QMR do not help to solve the problem of radiation introduced errors anymore and even increase the BER.

This behaviour can be explained by the shift in conductance. For higher radiation intensity more RRAM states tend toward higher conductance and are therefore recognized as another state. For lower radiation intensity this behaviour only appears rarely and the chance of misclassification of a state in the majority of the memory banks is low. However, for higher radiation intensity the chances drastically increase for the same state to be miss-classified in multiple banks at the same time.

On the other hand the power draw for the TMR and QMR mode is almost three or four times as high as the unprotected configuration while both are active. This is caused by the parallel usage of all three or four memory banks. However, we know from the analysis of solar events, that high intensity radiation events only occur rarely at about 2.5% of the time. Therefore, we take into account, that the fault-tolerant mode for the memory subsystem only

Table 3 Permutation Table for Bit-to-State mapping used for the extended TMR and QMR strategy

Mapping	00	01	10	11	Strategy
Map0	0	1	2	3	Base
Map1	1	0	3	2	TMR
Map2	3	2	1	0	TMR
Map3	2	3	0	1	QMR

needs to be enabled maximum 2.5% of the complete operation time. Once enabled a backup operation of the memory content from one single bank to all the other banks must follow. Using this knowledge we can make an estimation for the actual required power draw.

With this knowledge we can see in Table 2(b) and (c), that the power draw in the TMR configuration is only about $1.5\times$ higher than the unprotected configuration, while the QMR configuration is about $1.75\times$ higher. At the same time it can be seen that for our RRAM memory subsystem the QMR mode is for no radiation configuration more reliable than the TMR mode. Therefore the TMR mode seems to be the better candidate. For critical applications on the other hand both strategies are not enough, if a BER of near 0% needs to be guaranteed.

We can see that the TMR and QMR concept help to reduce the BER for low radiation intensity, but not completely mitigate all errors. At the same time for high radiation intensity these concepts do not provide a benefit anymore. We were able to trace this problem back to the overlap of the conductance distribution of the lower conductive state, which is degrading the TMR and QMR performance significantly. Programming multiple cells to the same state for redundant storage therefore does not result in a good reliability. However, we have already shown in a another study [33], that different mappings of data values to RRAM states can influence the system performance significantly. Therefore we introduce the extended TMR/QMR concept for RRAMs: we map the same data bits to different states in each of the RRAM memory banks that are used in parallel. Some states are more resilient to radiation than others and therefore with this configuration the individual miss-classification errors can cancel out. We have chosen a bit-to-state mapping that drastically alters, which bits are mapped to more vulnerable states for each memory bank. The permutation table for the base strategy, for TMR with three memory banks and for QMR with four memory banks can be seen in Table 3.

The resulting BER and power draw of the extended TMR and extended QMR strategy are listed in Table 2(d) and (e). The BERs of both strategies are also shown in Fig. 9. It can be seen that with the same energy overhead as the TMR and QMR strategy respectively (of about $1.5\times$ and $1.75\times$ compared to the baseline) the extended TMR strategy provides an almost perfect protection with a BER of about 0% for all radiation configurations up to R30. For R30 intensity the BER is reduced by nearly 50% and for R40 and R50 the BER is reduced by about 33% compared to the TMR strategy. Therefore, these strategies are a massive improvement, since they require almost no hardware overhead compared to the TMR and QMR strategies. In this configuration, the probability of incorrect classification is different in each RRAM device used in parallel, as the information is encoded differently everywhere. We had expected that this could potentially also help the majority voting with four states (QMR) to correct errors better than TMR. However, this is not the case. At this point, it should be noted that a promising improvement could be to perform weighted majority voting. There, the less susceptible states have a greater influence on the result. However, this requires a complete redesign of the majority voter and is therefore not pursued further here.

5.2 1-Bit RRAM cells

The extended TMR and QMR strategies provide a significant protection against low radiation intensity at the cost of a relative low power overhead. For higher radiation intensity on the other hand the resulting BER is still relatively high. There is however, one more strategy that can be applied to the RRAM cells to make them more radiation resilient. Instead of storing 2 bits or four states in a single cell only one bit can be stored. This avoids the overlap of neighbouring conductance levels under high radiation intensity. For this strategy two states which have a target conductance with a big distance between them can be chosen, such as L1 ($50 \mu S$) and L4 ($200 \mu S$) in our case. At the same time an ADC can be used, that accounts for the radiation effect. We have seen that radiation influences L1 far more and shifts its mean conductance strongly toward L4. L4 on the other hand is barely changed. Therefore an ADC threshold can be chosen, which is closer to the higher conductive state (L4 in our case).

The circuit design and performance model for our RRAM memory banks are based on the work of [32]. For this design, no changes to the circuit are required if only two states are stored instead of four. There is only a difference in the control, as only one instead of two comparison operations are performed during readout. This also halves the latency time for reading operations. The energy consumption for writing the values does not change on average. The reduction in energy consumption for reading is negligible. The only disadvantage of this solution is the increased power consumption due to the higher number of memory cells. As only half of the information is stored per cell, twice as many cells are required per memory bank. The resulting BER and power draw for this 1-Bit cell configuration is listed in Table 2(f). The BER is 0% for all radiation intensities, while the power consumption is about $2\times$ higher than the unprotected configuration. Therefore this configuration offers a perfect protection against radiation, while causing the highest power draw.

To combat the higher power draw a slight redesign of the TETRISC+RRAM architecture can be made. Since the average time in the fault-tolerant mode is only 2.5% of the overall runtime we can can introduce a dynamic switch between 2-Bit and 1-Bit cells. To implement this one could either (a) use a dynamic mapping, where in fault-tolerance mode of the memory subsystem, 2 Bit of data will be written into two separate RRAM cells. When not in fault-tolerance mode, 2 Bit of data will be written into one singe RRAM cell. However, this requires a control unit, which implements the mapping and a configurable ADC. Using this approach the available memory per bank would be effectively halved in fault-tolerance mode. The other possible implementation (b) would be to have two memory banks with 2-Bit RRAM cells and two memory banks with double the amount of RRAM cells, storing only 1-Bit each. When a radiation event is predicted the resilient 1-Bit cells need to be used. Before one backup operation is necessary and the power draw of the memory access doubles during that time. When the complete memory is used the overall power draw is about $1.5\times$ as high as the baseline. And in cases where fault-tolerance is not needed or two memory banks are enough the power draw is as high as for the unprotected configuration.

The results for this concept are listed in Table 2(g) as "extended 1-Bit Cells". Here the power draw is calculated using the assumption of a fault-tolerance mode on-time of 2.5% of the overall runtime. It can be seen, that this is a good configuration, providing complete reliability, with a low power footprint.

5.3 Applications

As discussed satellites are used for a variety of different applications. This means that different algorithms could be executed on the TETRISC+RRAM platform. In the following, we will therefore examine a representative sample of algorithms and evaluate their vulnerability to radiation-induced errors in the RRAM-based memory subsystem. We will use the BERs from the evaluation in the previous section to determine what protection strategies are suitable to guarantee a valid execution of different algorithms.

In order to target a wide number of applications and circumstances we opted to use the Embench benchmark suite for our simulated applications [34]. One of the main advantages over other free and open-source embedded benchmarks is the separation into different, self-contained applications for each benchmark case. This allows us to take many different algorithms into account without modifications to the source code. In Table 4 the implemented algorithms of Embench are listed along with some statistics about memory accesses. *Percentage Loads* denotes the proportion of memory load instructions to all instructions, while *Total Loads* represents the cumulative amount of memory accessed. The numerous algorithms are from different application domains, such as cryptography (aha-mont64, nettle-aes, nettle-sha256, md5sum), data structure traversal (tarfind, sglib-combined, huffbench), numerics (minver, cubic, st, matmult-int, nbody, ud), read and write of data formats (picojpeg, qrduino), and miscellanous other domains (crc32, primecount, wikisort, slre, edn, statemate, nsichneu). All of these algorithms could be part of a sattelite application in one form or another.

In addition to different algorithms we also take in to account different compiler optimizations. Higher levels of optimization typically result in fewer memory accesses, which can ultimately lead to improved resilience under radiation. Additionally, we benchmark the algorithms with the option for position-independent executables (PIE) enabled and disabled. Although, position-independent executables are the default setting for most RISC-V compiler suites, they can potentially lead to additional memory accesses when referencing global variables. This can have a slight effect on the memory access patterns and therefore impact the chance of a correct program execution. The compiler can produce code accessing global memory in some rare cases via an additional data structure in runtime memory, which leads to additional read accesses to memory.

For the simulation fault injection was limited to program sections of the algorithm under benchmark. The verification of the results and any setup and loading remained unaffected. Consequently, the effect of faults only refers to the algorithms themselves and not the operating system. This approach was chosen, because if a memory error occurs during loading of a program, the likelihood of a program crash is very high and these failures cannot be attributed to the algorithm under consideration. For a real world application it is necessary to ensure the correct loading of an executable with extra protection.

We have evaluated the behavior of all algorithms implemented in the Embench benchmark suite under radiation. For this purpose, we examined all possible combinations of four different compiler optimization levels ("-O0", "-O2", "-O3", "-Ofast", in increasing order) and the option to produce PIE. For the simulation of the radiation effect, we examined the results for the unprotected memory subsystem, protection with extended TMR, and protection with extended QMR, respectively. We did not simulate the protection concept with 1-bit RRAM cells separately, since this corresponds to the behavior of the system not influenced by radiation due to the BER of 0%. Each benchmark case was run 512 times for each BER and compiler configuration. Due to the amount and complexity of the data an overview can be found in the "Appendix 1" of this article.

Table 4 Algorithms implemented in the Embench benchmark suite and memory access statistics

	Description	Percentage loads (%)	Total loads (MB)
aha-mont64	Montgomery Multiplication, 128-bit arithmetic	0.13	0.020
minver	Floating Point Matrix Inversion	3.81	0.040
cubic	Cubic Polynomial Solver	5.56	0.457
tarfind	Search in TAR archive	6.48	0.338
st	Statistic Algorithms	8.85	0.083
crc32	Checksum Algorithm	9.09	2.785
md5sum	Hashing Algorithm	9.16	1.164
nbody	Floating Point N-Body Problem	9.93	0.001
nettle-sha256	Cryptographic Hashing Algorithm	10.35	2.250
picojpeg	JPEG Decode	14.41	1.405
huffbench	Huffman Compression	15.88	1.990
nettle-aes	Encryption Algorithm	16.25	3.080
qrduino	QR Code Generator	16.71	1.430
wikisort	Sorting Algorithm	18.51	0.374
primecount	Prime Sieve	18.86	1.502
slre	Regular Expressions	19.46	3.099
matmult-int	Integer Matrix Multiplication	22.07	6.330
sglib-combined	Data Structure Benchmark (Array, Linked List, Hash Table, Queue, RB Tree)	23.51	3.100
edn	MDH WCET Benchmark Suite (Vector Multiplication, FIR and IIR Filters, Lattice Synthesis, Vocoder Codebook Search, JPEG Discrete Cosine Transform)	25.63	1.910
ud	LU Decomposition	30.09	2.270
statemate	Electric Window Control	32.27	0.868
nsichneu	Pr/T-Net Execution	54.82	8.568

In general, the following observations can be made for the algorithms under the influence of the radiation induced bit errors from the RRAM memory subsystem:

- Algorithms with low memory access rate are generally more robust toward errors introduced to the RRAM-based memory subsystem by low radiation intensity.
- Higher levels of compiler optimization are generally advantageous for ensuring the accurate execution of an algorithm. However, the extent of the impact of these optimizations varies greatly, depending on the specific algorithm being evaluated.
- Without compiler optimizations, most applications are likely to experience significantly low success rates of execution.
- For certain algorithms, disabling the PIE feature yields a beneficial outcome.
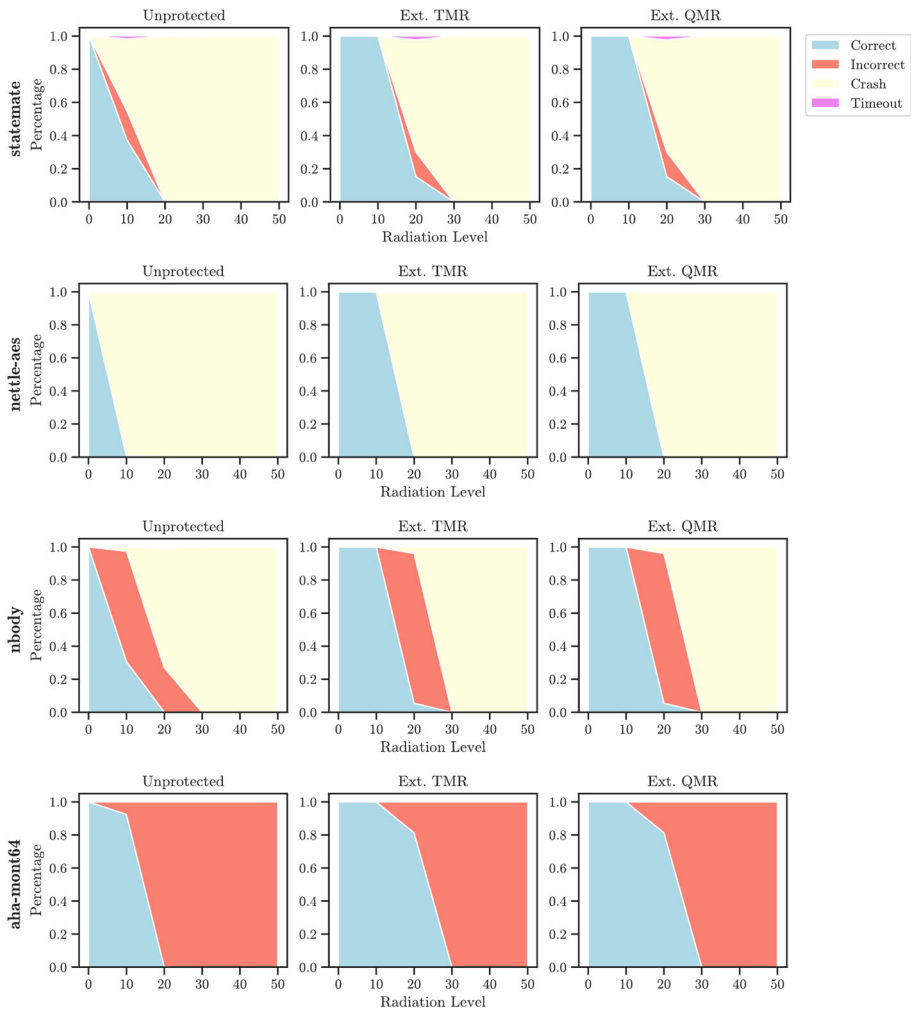
Fig. 10 Execution result of selected algorithms in relation to the radiation levels

- The 'nsichneu' algorithm represents a special case of program flow. If its main loop terminates early due to memory errors, the program run initially appears correct, since the state variables match those of a normal run. In such cases, the algorithm result is classified here as 'correct', although the operation performed is incomplete.

We selected four representative algorithms from the Embench benchmark suite ('statemate', 'nettle-aes', 'nbody' and 'aha-mont64') and visualized the results of their simulation. These results were generated in the configuration with the highest compiler optimization level ("-Ofast") enabled and the option for PIE turned off. Figure 10 shows an overview of these results for the unprotected system, the extended TMR and the extended QMR strategy. It is striking that even at low doses of radiation, the rate of correct execution of all algorithms drops extremely sharply in all three cases. This clearly shows that many classical algorithms are not very robust against bit errors and already get problems with small deviations of the

memory content. It can also be seen that for most algorithms the dominant error case is the crash of the program. On the one hand, this is an error case that is easily detected by an operating system and occurs almost immediately. On the other hand, it makes it difficult to guarantee a successful run of an application even with protection under higher radiation exposure. The 'aha-mont64' algorithm has a unique design with minimal memory access and numerous mathematical operations. Here it is shown that such algorithms are generally more robust against the influence of higher radiation intensities, since they often deliver a correct result even at higher intensities. Furthermore, even under intense radiation, the algorithm remains functional without crashing. However, a correct result of the algorithm is then almost impossible. In a real application, this means that the problems caused by radiation are difficult or impossible to detect and can cause serious problems later. While the extended TMR and extended QMR strategy significantly lower the BER, it is shown for the applications that this protection is not directly transferable to the correct execution of the algorithms. While they achieve better protection at low radiation levels in most cases, there is still a very high chance of a crash of the application. Another notable finding is that algorithm timeouts hardly play a role. They only occur very rarely under lower radiation intensity. The frequency of errors in the execution of algorithms is far exceeded by cases in which the algorithms work correctly and those in which they produce incorrect results or cause a crash.

6 Analysis and discussion

We have introduced the TETRISC+RRAM platform with a new designed RRAM-based memory subsystem, that can dynamically switch between fault-tolerant, power-saving and high-performance mode. An evaluation of possible radiation resilience strategies to protect the system from radiation induced errors has shown how badly the memory content is affected by the radiation in form of the BERs for different intensities. The evaluation of the behaviour of different algorithms on the platform under radiation showed the resilience requirement of different application scenarios. In the following we will discuss how a real world implementation and usage of the system could look like.

Analyzing the BER and power draw of the system, it can be seen, that for the RRAM-based memory subsystem the TMR strategy is superior to the QMR strategy, since the resulting BER is almost identical, while at the same time the TMR strategy requires a lower power draw. For the final system configuration it would therefore be beneficial to only implement a programmable TMR voter for the memory. The analysis of the BER also shows that architectural parameters, such as the mapping of bits to RRAM states have huge influence on the systems overall performance. The extended TMR configuration is an easily implementable option to significantly reduce the BER of the memory system even further.

When analyzing the relationship between radiation levels and the behavior of applications, it is shown that especially memory intensive applications are very hard to protect against radiation using the extended TMR or extended QMR strategy. On the other hand algorithms that can be highly optimized to rarely access the main memory, are nearly independent of the BER and can run on the system under high radiation without significant protection (e.g. such as the extended TMR strategy), but a prone to produce erroneous results.

An additional point of interest is the way applications tend to fail: In most cases there is first an increase in undetected errors withing the calculation result and with rising radiation levels most applications just crash. It is highly dependent on the use case if this behavior is favourable. For some cases a calculation error is intolerable and a crash is therefore a more
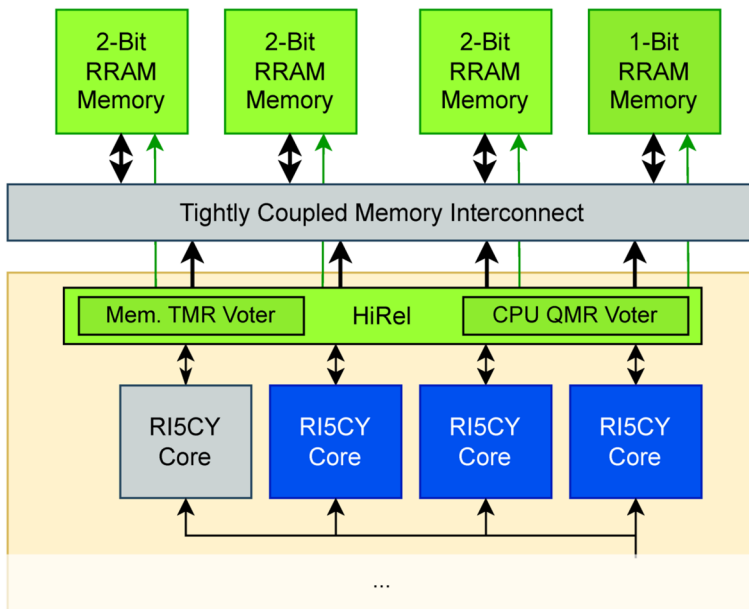
Fig. 11 Excerpt of the block diagram of final proposed system configuration

favourable outcome. On the other hand, in some applications it might be acceptable to have some degree of errors and the results are still acceptable to the user and an improvement over a simple crash. For example artificial neuronal nets can often tolerate some degree of faulty calculations, while most cryptographic algorithms are very susceptible and will yield a totally unusable result. Both can be an application scenario for a satellite, depending on the task at hand.

In isolated instances, algorithms may not terminate due to certain error conditions. For algorithms showing this behaviour, it might be necessary to implement a watchdog functionality to detect and interrupt the execution of these algorithms. However, for most cases we have evaluated, this effect is eclipsed by application crashes, which are a likely outcome during the increased runtime. The advantage of the crash is that it can be detected more easily and a rerun of the algorithm can be triggered.

Another problem can be undetected false results of an application under radiation. However, here again an invalid internal state might not manifest in an erroneous output, if a crash occurs before the end of the program. And for most algorithms crashes will dominate as the failure mode.

With this knowledge of the in depth evaluation of the memory subsystem under radiation and the behaviour of applications we suggest an adapted system design independent of the application. An excerpt of the block diagram of this final proposed design is shown in Fig. 11. We advise for a solution with a programmable TMR voter for the memory, together with three RRAM-based memory banks using 2-Bit RRAM devices and one memory bank using 1-Bit RRAM devices. This configuration provides a broad flexibility. During normal operation in performance or low-power mode, the 2-Bit RRAM memory banks can be used. For less critical algorithms these three memory banks can be configured for extended TMR mode using the memory majority voter. This will provide a significant protection, but might result

in algorithm crashes or incorrect results, during high intensity radiation. The 1-Bit RRAM memory bank can be used during a radiation event for highly critical algorithms, with high memory access and basic system functionality, such as program loading. This will ensure the correct functionality also under harsh radiation such as during a SPE. Depending on the specific application further optimizations can be implemented, such as the usage of the unprotected 2-Bit memory banks during a radiation event, for extremely uncritical or error tolerant algorithms.

7 Conclusion

In this work we have shown for the first time a RRAM-based memory subsystem design for satellites. It can dynamically switch between performance, power-saving and fault-tolerance mode. We have iteratively shown which radiation protection concepts can be applied for this system to cope with radiation induced bit errors and how actual algorithms executed on the system behave in these radiation conditions. Therefore, we could define optimal operation conditions and in the end suggest an implementation of this subsystem.

One area of further investigations can be the radiation model. Until now an extrapolated model is implemented which mainly focuses on the shift of the mean conductance levels. Once actual measurements under radiation for the RRAM devices are available, a more in depth characterization can yield better insights on behaviour such as the change of the variance of the distribution, which might have a significant influence on the BER. At the same time different radiation levels can be more precisely correlated with the change in conductance of the RRAM devices.

A further extension of the TETRISC+RRAM platform could be the development of a weighted majority voting strategy, where those RRAM states, which are less affected by radiation have a bigger influence on the outcome. Additionally, the TETRISC+RRAM platform can be further extended, by enabling in-memory computing functionality of the RRAM devices inside the memory banks. This can lead to massive performance improvement and power draw reduction, for algorithms that can be mapped to in-memory computing operations.

Declarations

Conflict of interest The authors do not declare any Conflict of interest or Conflict of interest.

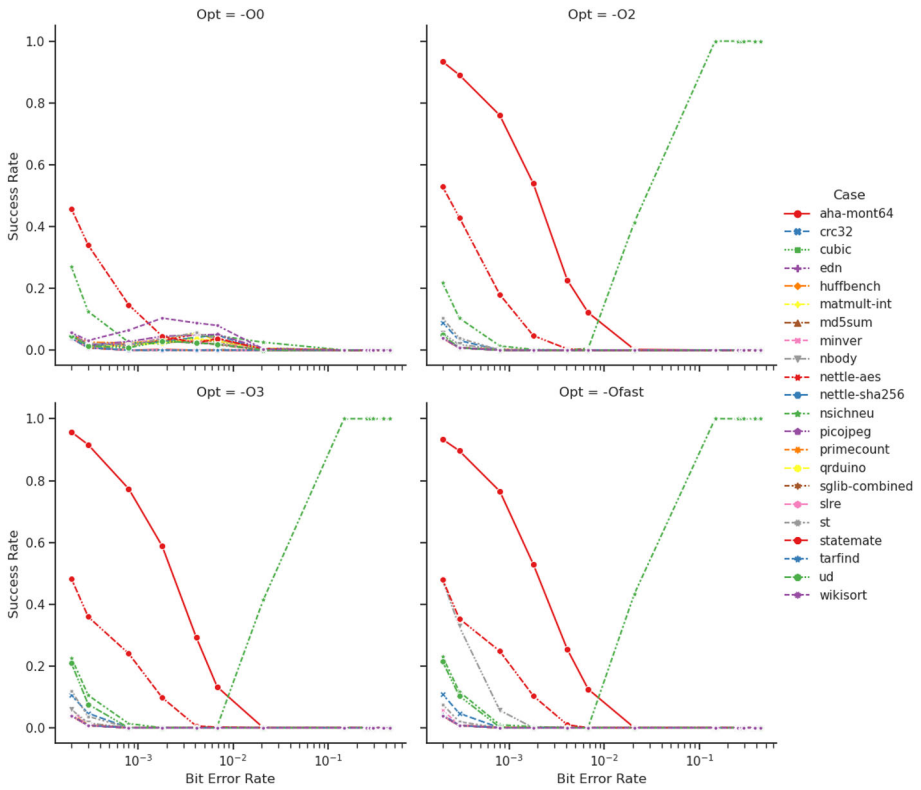# Appendix A simulation data

See Figs. 12 and 13.

Fig. 12 Results of simulating all Embench benchmark suite algorithms under radiation with PIE enabled
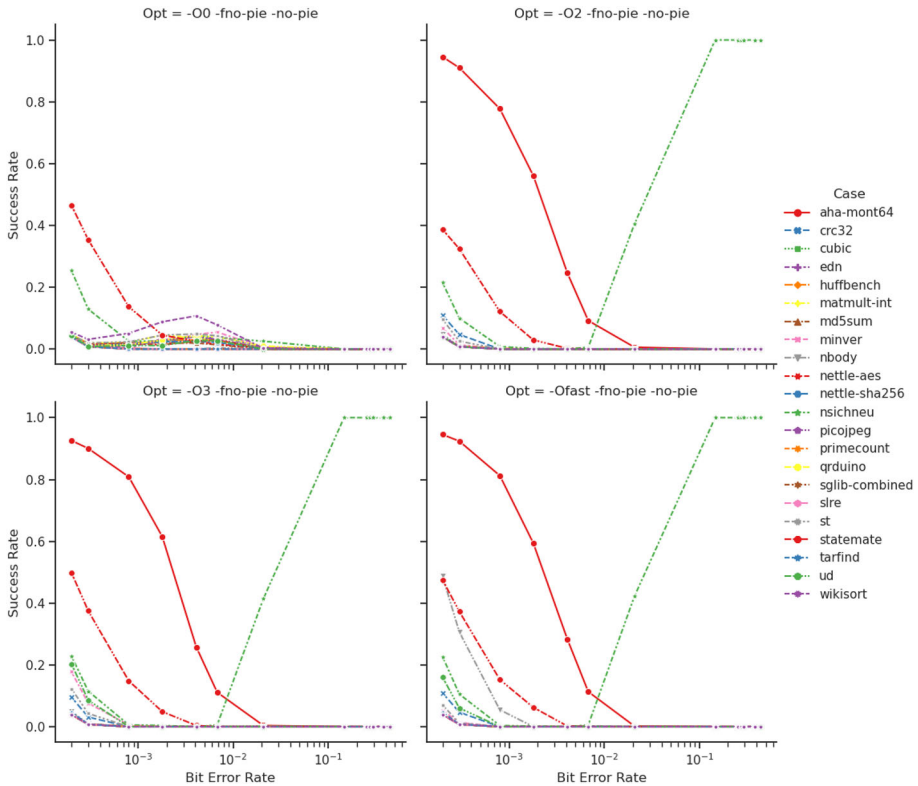
Fig. 13 Results of simulating all Embench benchmark suite algorithms under radiation with PIE disabled

References

1. Euroconsult. Space and satellite sector expert. https://www.euroconsult-ec.com/
2. Denby B, Lucia B (2020) Orbital edge computing: nanosatellite constellations as a new class of computer system. In: Proceedings of the twenty-fifth international conference on architectural support for programming languages and operating systems. ASPLOS '20. New York, NY, USA: Association for Computing Machinery, pp 939–954. https://doi.org/10.1145/3373376.3378473
3. Bourdarie S, Xapsos M (2008) The near-earth space radiation environment. IEEE Trans Nucl Sci 55(4):1810–1832. https://doi.org/10.1109/TNS.2008.2001409
4. Safari S, Ansari M, Khdr H, Gohari-Nazari P, Yari-Karin S, Yeganeh-Khaksar A et al (2022) A survey of fault-tolerance techniques for embedded systems from the perspective of power, energy, and thermal issues. IEEE Access 10:12229–12251. https://doi.org/10.1109/ACCESS.2022.3144217
5. Mushtaq H, Al-Ars Z, Bertels K (2011) Survey of fault tolerance techniques for shared memory multi-core/multiprocessor systems. In: 2011 IEEE 6th international design and test workshop (IDT), pp 12–17
6. Zhang B, Wu Y, Zhao B, Chanussot J, Hong D, Yao J et al (2022) Progress and challenges in intelligent remote sensing satellite systems. IEEE J Sel Top Appl Earth Obs Remote Sens 15:1814–1822. https://doi.org/10.1109/JSTARS.2022.3148139
7. Gonzalez-Velo Y, Barnaby HJ, Kozicki MN (2017) Review of radiation effects on ReRAM devices and technology. Semicond Sci Technol 32(8):083002. https://doi.org/10.1088/1361-6641/aa6124
8. Chen J (2023) A self-adaptive resilient method for implementing and managing the high-reliability processing system [doctoral dissertation]. Universität Potsdam
9. Ulbricht M, Lu L, Chen J, Krstic M (2023) The TETRISC SoC-A resilient quad-core system based on the ResiliCell approach. Microelectron Reliab 148:115173. https://doi.org/10.1016/j.microrel.2023.115173

10. Barth JL, Dyer CS, Stassinopoulos EG (2003) Space, atmospheric, and terrestrial radiation environments. IEEE Trans Nucl Sci 50(3):466–482. https://doi.org/10.1109/TNS.2003.813131

11. GOES. Geostationary operational environmental satellites—space environment monitor database. https://ngdc.noaa.gov/stp/satellite/goes/dataaccess.html

12. National Oceanic and Atmospheric Administration. Solar proton events affecting the earth environment lists. https://www.ngdc.noaa.gov/stp/satellite/goes/doc/SPE.txt

13. Harboe-Sorensen R, Daly E, Teston F, Schweitzer H, Nartallo R, Perol P et al. (2001) Observation and analysis of single event effects on-board the SOHO satellite. In: RADECS 2001. 2001 6th European conference on radiation and its effects on components and systems (Cat. No. 01TH8605), pp 37–43

14. Yearby KH, Balikhin M, Walker SN (2014) Single-event upsets in the cluster and double star digital wave processor instruments. Space Weather 12(1):24–28. https://doi.org/10.1002/2013SW000985

15. Chen J, Lange T, Andjelkovic M, Simevski A, Lu L, Krstic M (2022) Solar particle event and single event upset prediction from SRAM-based monitor and supervised machine learning. IEEE Trans Emerg Top Comput 10(2):564–580. https://doi.org/10.1109/TETC.2022.3147376

16. Bennett WG, Hooten NC, Schrimpf RD, Reed RA, Mendenhall MH, Alles ML et al (2014) Single- and multiple-event induced upsets in HfO_2/Hf 1T1R RRAM. IEEE Trans Nucl Sci 61(4):1717–1725. https://doi.org/10.1109/TNS.2014.2321833

17. Chen D, LaBel KA, Berg M, Wilcox E, Kim H, Phan A et al. (2014) Radiation effects of commercial resistive random access memories. In: NASA electronic parts and packaging (NEPP) electronics technology workshop (ETW). GSFC-E-DAA-TN16279

18. Song H, Ni K, Tang Y, Wang J, Guo H, Zhong X (2021) Total ionizing dose effects of 60 Co-y ray radiation on the resistive switching and its bending performance of Al-in-O/InOx-based flexible RRAM device. Radiat Phys Chem 182:109394. https://doi.org/10.1016/j.radphyschem.2021.109394

19. Barella M, Sanca G, Marlasca FG, Acevedo WR, Rubi D, Inza MAG et al (2019) Studying ReRAM devices at low earth orbits using the LabOSat platform. Radiat Phys Chem 154:85–90. https://doi.org/10.1016/j.radphyschem.2018.07.005

20. Wang Y, Lv H, Wang W, Liu Q, Long S, Wang Q et al (2010) Highly stable radiation-hardened resistive-switching memory. IEEE Electron Device Lett 31(12):1470–1472. https://doi.org/10.1109/LED.2010.2081340

21. Ma Y, Yang X, Bi J, Xi K, Ji L, Wang H (2021) A radiation-hardened hybrid RRAM-based non-volatile latch. Semicond Sci Technol 36(9):095009. https://doi.org/10.1088/1361-6641/ac117b

22. Tosson AMS, Yu S, Anis MH, Wei L (2018) Proposing a solution for single-event upset in 1T1R RRAM memory arrays. IEEE Trans Nucl Sci 65(6):1239–1247. https://doi.org/10.1109/TNS.2018.2830791

23. Leitersdorf O, Perach B, Ronen R, Kvatinsky S (2021) Efficient error-correcting-code mechanism for high-throughput memristive processing-in-memory. In: 58th ACM/IEEE design automation conference (DAC), pp 199–204

24. Jacobs A et al (2012) Reconfigurable fault tolerance: a comprehensive framework for reliable and adaptive FPGA-based space computing. ACM Trans Reconfig Technol Syst 5(4):21:1-21:30

25. Glein R et al (2018) Adaptive single-event effect mitigation for dependable processing systems based on FPGAs. Microprocess Microsyst 59:46–56

26. Ferreira RS, Nolte J, Vargas F, George N, Hübner M (2020) Runtime hardware reconfiguration of functional units to support mixed-critical applications. In: 2020 IEEE Latin-American test symposium (LATS), pp 1–6

27. Lukefahr A et al (2012) Composite cores: pushing heterogeneity into a core. In: 2012 45th annual IEEE/ACM international symposium on microarchitecture, pp 317–328

28. Chen J, Andjelkovic M, Simevski A, Li Y, Skoncej P, Krstic M (2019) Design of SRAM-based low-cost SEU monitor for self-adaptive multiprocessing systems. In: 2019 22nd Euromicro conference on digital system design (DSD), pp 514–521

29. Chen J, Lange T, Andjelkovic M, Simevski A, Krstic M (2020) Hardware accelerator design with supervised machine learning for solar particle event prediction. In: 2020 IEEE international symposium on defect and fault tolerance in VLSI and nanotechnology systems (DFT), pp 1–6

30. Simevski A, Hadzieva E, Kraemer R, Krstic M (2012) Scalable design of a programmable NMR voter with inputs' state descriptor and self-checking capability. In: 2012 NASA/ESA conference on adaptive hardware and systems (AHS), pp 182–189

31. Zambelli C, Grossi A, Olivo P, Walczyk D, Bertaud T, Tillack B et al (2014) Statistical analysis of resistive switching characteristics in ReRAM test arrays. In: 2014 international conference on microelectronic test structures (ICMTS), pp 27–31

32. Pechmann S, Mai T, Potschka J, Reiser D, Reichel P, Breiling M et al (2021) A low-power RRAM memory block for embedded, multi-level weight and bias storage in artificial neural networks. Micromachines 12(11):1277

33. Reiser D, Reichenbach M, Rizzi T, Baroni A, Fritscher M, Wenger C et al (2023) Technology-aware drift resilience analysis of RRAM crossbar array configurations. 2023 21st IEEE interregional NEWCAS conference. https://doi.org/10.1109/NEWCAS57931.2023.10198076
34. Bennett J. Embench: open benchmarks for embedded platforms