Reliability evaluation of Convolutional Neural Network's basic operations on a RISC-V processor

Fernando Fernandes dos Santos, Angeliki Kritikakou, Olivier Sentieys

Reliability evaluation of Convolutional Neural Network's basic operations on a RISC-V processor

Fernando Fernandes dos Santos, Angeliki Kritikakou, and Olivier Sentieys

***Abstract*—Thanks to RISC-V open-source Instruction Set Architecture, researchers and developers can efficiently propose new solutions at a low-cost and low-power consumption. RISC-V-based architectures can, then, be customized to run Machine Learning (ML) algorithms efficiently and be inserted on safety and mission-critical domains, where the execution must be reliable. However, a fault in the hardware resources can compromise the system's ability to operate correctly. Thus, it is necessary to characterize the ML applications' vulnerabilities on RISC-V processors, as it is not as thoroughly characterized as other accelerators. This work is the first to evaluate the neutron-induced error rate of Convolutional Neural Network (CNN) basic operations running on a RISC-V processor, GAP8. Our results show that executing the algorithm in parallel increases performance, and memory errors are the major contributors to the device error rate. We show that the error rate of the GAP8 unprotected memories is one order of magnitude higher than the CNN basic operations.**

I. Introduction

Recent advances in Machine Learning (ML) algorithms, such as quantization, reduced precision training, and weight pruning, enabled tiny Convolutional Neural Networks (CNNs) models to be adopted in low-power consumption processors and accelerators, such as TPUs, tiny ARM CPUs, and RISC-V SoCs. Compared to other architectures, RISC-V processors have the advantage of implementing an open-source Instruction Set Architecture (ISA), allowing designers to propose new customized architectures with smaller non-recurring engineering costs. RISC-V processors are today adopted in several domains, including end-user applications [1], High-Performance Computing (HPC) [2], and safety-critical applications [3], [4]. Such a market trend became a promising option for CNN-based safety-critical applications where power consumption, real-time execution, and reliability are mandatory.

Researchers have been overly focused on improving CNN's performance and power consumption on recent RISC-V architectures [5], [6]. However, for RISC-V processors to be employed on safety-critical applications, their reliability must be thoroughly characterized to define how faults can impact programs and prevent the system from fulfilling the mandatory constraints. Faults that disrupt a system's operation can be generated by different events, such as environmental perturbations, ionizing radiation, software errors, process, temperature, or voltage variations [7], [8]. Ionizing radiation is particularly dangerous for safety-critical applications as it leads to very high error rates [9]. Terrestrial neutrons can perturb the state of a transistor or change memory cells' values, leading to soft errors. Note that the device is not permanently damaged (a new operation or memory write will replace the faulty value), but the error may propagate to the output.

To evaluate a device radiation-induced error rate, researchers perform radiation experiments with CPUs [10], GPUs [11]–[13], FPGAs [14], [15], and Tensor Processing Units [16]. The error rate of soft RISC-V processors synthesized on FPGAs, or ASICs, has been measured when exposed to heavy ions and neutrons [17]–[20]. In this paper, we go above the state-of-the-art by not only evaluating the error rate of an ASIC RISC-V processor but also performing a detailed reliability evaluation of the most basic CNNs operations on a Parallel Ultra Low Power (PULP) RISC-V System on Chip (SoC), GAP8 from GreenWaves [5]. GAP8 includes an 8-cluster core that supports standard RISC-V instructions and a set of Single Instruction Multiple Data (SIMD) instructions. SIMD instructions are extremely interesting for CNNs as they can improve performance without increasing power consumption.

We expose the GAP8 on a neutron beam and measure its error rate while running the main CNNs operations (Convolution, Fully connected Linear, and Maxpool). We report findings from recently completed neutron beam testing, which represents a total of more than 130,000 years of operation in the terrestrial environment. As CNNs are both computationally- and memory-demanding and GAP8 does not have memory error protection, we evaluate the error rate and criticality of the main GAP8's memories. We discuss how memory errors can impact the main CNNs operations error rate.

The remainder of this paper is as follows. In the next Section, we present the background on radiation effects on electronic and RISC-V-based processors. Section III describes the methodology used to measure GAP8's realistic error rate. The experimental measured error rate and main results are presented in Section IV. Section V concludes the paper.

II. Radiation effects on RISC-V devices

Terrestrial high-energy neutrons that interact with the hardware may generate soft errors in the system. The striking particle can generate single or multiple-bit flips on a memory resource, such as caches, registers, or buffers, or even corrupt the value of a functional unit inside a processing core. If the value is used as part of the algorithm computation, the incorrect value will be propagated by the code running on the device. At the application level, the failure may manifest as the following outcomes: (1) **No effect on the program**

Fernando Fernandes dos Santos, Angeliki Kritikakou, and Olivier Sentieys are with Univ. Rennes, INRIA, Rennes, France
Contact e-mail: {fernando.fernandes-dos-santos, angeliki.kritikakou, olivier.sentieys}@inria.fr

TABLE I: Characteristics for the evaluated microbenchmarks on GAP8 RISCV (memory and CNN).

	SoC usage	Cycles [KB]	Mem size
Linear	Sequential	57.6	1024x16 bytes
	Parallel+Vector	2.2	
Maxpool	Sequential	25.7	112x112 bytes
	Parallel+Vector	3.8	by 2x2/2 filter
Convolution	Sequential	913.4	112x112 bytes
	Parallel+Vector	22.8	by 5x5 filter
Mem. L1	Sequential	–	62KB
Mem. L2	Sequential	–	448KB

output: The fault is masked, the program output is not affected or the circuit functionality is not modified. (2) **Silent Data Corruption (SDC)**: The program finishes, but the output is not correct, and neither flag nor indication is raised. (3) **Detected Unrecoverable Error (DUE)**: The system stops working, forcing it to be rebooted or power cycled. A DUE can result from uncorrectable memory events, crashes, or an error that generates an infinite loop (program hangs).

Prior works have studied the reliability of RISC-V cores synthesized on FPGAs with neutrons and heavy ions experiments [17]–[19]. As many RISC-V processors share an open-source concept, developers can modify the architecture to apply Full or Partial Modular Redundancy to increase the processor reliability and evaluate the hardened architecture on beam experiments [17], [21]. Recently, a commercial RISC-V processor's error rate and criticality have been characterized on a neutron beam experiment to determine the impact of neutron-induced faults on a small set of applications [20]. However, none of the prior works have focused the analysis on crucial ML algorithms for CNNs. In this paper, we point to two significant contributions: (1) We evaluate the most basic CNN operations running on an ASIC commercial multicore RISC-V platform (GAP8) on a neutron beam and extract the realistic cross-section; (2) We measure the memory cross-section and fault model of the main GAP8 memories and correlate the results with the cross-section observed on CNNs operations. This analysis provides valuable insights into the contributions of memory errors in the application error rate running on commercial RISC-V processors.

III. Metrics and Evaluation Methodology

In this Section, we describe the beam experiments methodology for measuring GAP8 RISC-V SoC's cross-section, the codes we evaluate, and the metrics we use to support the evaluation in the results section.

A. Device under test and evaluated codes

Device Under Test (DUT): For the beam experiments, we consider a commercial RISC-V SoC named GAP8. GAP8 is a multicore RISC-V platform from GreenWaves technologies based on PULP architecture. GAP8 is built with $55nm$ TSMC 55LP CMOS technology, and it consists of a cluster of 8 RISC-V cores connected by a Logarithmic Interconnect and a RISC-V Fabric Controler (FC) processor to manage the cluster. The FC core operates at 250MHz, while each cluster core operates at 175MHz. The FC core also has 16KB of data and 1KB of instruction cache. Both the FC and the cluster can access an L1 memory of 64KB and an L2 memory of 512 KB. None of the SoC memories have an Error Correction Code (ECC) nor protection against Single Events Upsets. GAP8 supports integer and fixed-point arithmetic. During our experiments, we set the voltage and frequencies of GAP8 SoC to the maximum optimal values according to the GAP8 manual (1.2 volts with 250MHz for the FC and 175MHz for the cluster). This configuration set has a power consumption of 75mW [5].

```
1  /**Memory test code**/
2  while no memory error is observed {
3      /**Set all the memory with AAAAAAAA**/
4      for i in every 4 bytes of the memory {
5          M[i] ⇐ 0xAAAAAAAA
6      }
7      sleep for 1s
8      /**Compare the memory values**/
9      errors ⇐ 0
10     for i in every 4 bytes of the memory {
11         errors += (M[i] != 0xAAAAAAAA)
12     }
13     /**If there is error(s)
14     return to the host and log**/
15     if errors != 0
16         break & log on the host
17 }
18
```

Fig. 1: Behavioral algorithm for L1 and L2 memory test.

Evaluated codes: We focus our analysis on the reliability of the core operations of CNNs. We perform two types of experiments where we characterize the memories and the most common operations in a CNN.

1) Memory microbenchmark:

As CNNs are known for being resource-demanding in terms of memory and computational resources, and as GAP8's main memories do not have ECC, we investigate the GAP8's memory cross-section and correlate it with the results for CNN operations. We design a microbenchmark to measure L1 and L2 memories cross-section. Figure 1 depicts a behavioral code for the memory microbenchmark. The code consists of a loop that runs while no memory errors are observed (line 2). The L1 or L2 memories of GAP8 are populated with a specific pattern (chunks of 4 bytes with $0xAA$) (lines 4 to 6). After the writing, the code sleeps for 1s (line 7). Finally, the microbenchmark reads all the memory values and logs on the host if there are errors (lines 9 to 16). The memory microbenchmarks use the maximum amount of memory possible, 87.5% of L1 SoC memory and 96.8% of L2 SoC memory.

2) Common CNN operations:

- *Convolution layer* is the main operation performed on a CNN. The selected code consists of a 5×5 filter convoluted into a matrix of 112×112. The Convolution layers also have the characteristics to be the most resource-demanding procedure of a CNN. Consequently, convolution layers have been demonstrated to be one of the most critical parts of CNNs [22]–[24]. As discussed in Section IV, the convolution operation has the highest error rate.
- *Fully connected Linear layer* is the class of algorithm that performs the last step of the inference on a CNN. The standard

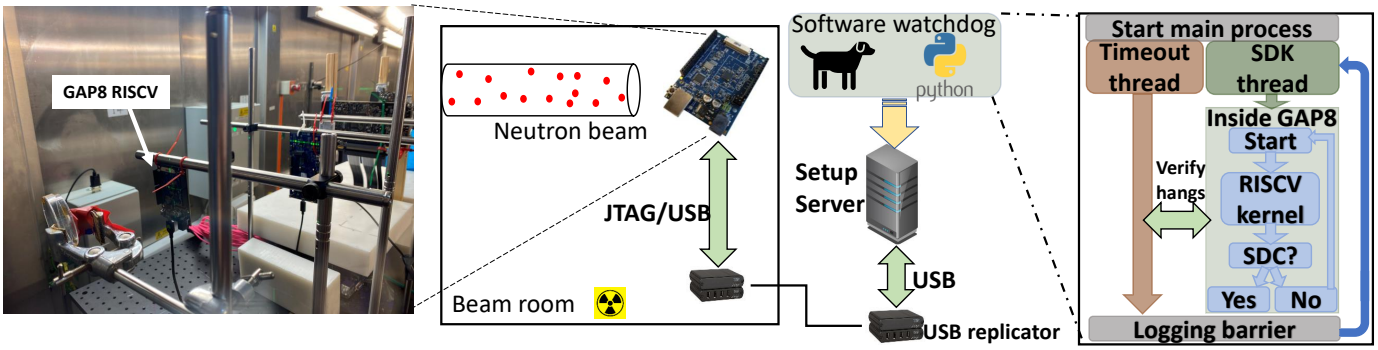

Fig. 2: ChipIR setup overview. The GAP8 is exposed to the neutron beam. The communication is performed through JTAG/USB by a fast USB replicator. Outside the beam room, a software watchdog server monitors the events inside the SoC.

fully connected neural networks receive input and, based on the neuron's activation, produce an inference that is reduced to a smaller set of values. In our experiments, the Linear layer operates over an input of 1024 bytes and produces 16 values, which correspond to probability values in a real CNN.

- *Maxpool layer* is a specific ML algorithm explicitly created to avoid over-fitting on CNNs. Maxpool is placed after a certain number of layers to reduce the amount of data by filtering the values based on a filter block. We selected a maxpool layer that applies a 2×2 filter to a matrix of size 112×112. For each 2×2 block, only the highest value is propagated. As proposed by prior works [22]–[24], maxpool is expected to mask at least 3/4 of the errors.

The selected codes are specially developed for GAP8 by Flamand *et. al.* [5]. The codes are designed to extract the best performance from the SoC and to compare the different implementations of the same algorithm. Each configuration changes regarding the number of cores and the usage or not of special SIMD vector instructions. In this work, due to beam time limitations, we select two configurations for each code, i.e., the least performance (using only one core of the GAP8) and the highest performance (using all eight cores and SIMD vector instructions).

B. Beam Experiment Setup

In order to effectively evaluate the reliability of GAP8, we expose it to a neutron beam and measure its cross-section. Similar to [5], [20], we use the GreenWaves Software Development Toolkit (SDK) to build and run the experiments. Our experiments are performed at the ChipIR facility of the Rutherford Appleton Laboratory, UK. The available neutron flux was about $3.5 \times 10^6 n/(cm^2/s)$, ~8 orders of magnitude higher than the terrestrial flux at sea level [25]. The facility delivers a beam of neutrons with a spectrum of energies that resembles the atmospheric neutron one [26]. As a metric to characterize the error rate, we calculate the *cross-section* by dividing the number of observed errors ($\#errors$) by the received particle fluence (η), i.e., Equation 1. The fluence is obtained by multiplying the average neutron flux of the test facility ($neutrons/(cm^2 \cdot s)$) by the effective code execution time in seconds.

$$\sigma[cm^2] = \frac{\#errors}{\eta} \quad (1)$$

The cross-section (cm^2) represents the circuit area that will generate an output error (SDC or DUE) if hit by a particle. The higher the number of computation resources, the higher the cross-section, and the higher the probability of an impinging particle generating an error.

Figure 2 shows an overview of the experiments. We created a software watchdog consisting of Python scripts that runs on the setup server computer outside the beam room. The watchdog controls GAP8 by monitoring, executing the programs, logging events, and recovering from device hangs. The program is killed and relaunched if it stops responding in a predefined interval. We set each code timeout individually depending on the code execution time, up to $3\times$ the expected execution time. The code inside GAP8 executes the same kernel for a predetermined number of iterations. We set the maximum number of internal SoC iterations to 65KB for a good tradeoff between time wasted with communication and proper kernel executions. After each iteration inside the SoC, the output of the kernel is compared with a constant golden value. If there is a mismatch between both outputs, the main process will log the useful information and restart again. It is worth noting that errors only errors that happen on the kernel output are considered for the error rate analysis.

IV. GAP8 Cross-Section

We start our analysis by presenting the error rate observed on the memory microbenchmark. It paves the evaluation for the CNNs-based operation codes. We then compare the cross-section of various CNN operations and the differences between sequential and parallel algorithms. All results (CNN common operations and memory benchmark) are reported with 95% confidence intervals considering a Poisson distribution.

A. Memory cross-section

Table II shows the L1 and L2 microbenchmarks SDC cross sections observed on the beam experiments. We present the total cross-section that considers the cross-section of the entire evaluated memory (based on the memory sizes from Table I) and the cross-section per byte.

The memory cross-section per byte is similar to values measured in prior works with similar technologies [27], [28]. L1 cross-section per byte is 1.28×10^{-14} and L2 cross-section

TABLE II: Memory SDC Cross Section observed on the neutron beam experiments. L1 and L2 memories are evaluated.

	Total cross-section	Byte cross-section
L1	$8.13 \times 10^{-9} \pm 7.4 \times 10^{-10}$	$1.28 \times 10^{-13} \pm 1.2 \times 10^{-14}$
L2	$4.21 \times 10^{-8} \pm 3.9 \times 10^{-9}$	$9.18 \times 10^{-14} \pm 8.6 \times 10^{-15}$

per byte is 9.18×10^{-15}. Although L1 and L2 are similar memories, the L2 cross-section per byte is 29% smaller than the L1 cross-section. This is because they are differently organized on the SoC and, consequently, have different latencies and bus connections. More precisely, L1 is a shared memory between the cluster, while L2 is a larger memory divided into 4 blocks.

Figure 3 depicts the Single-Bit Upsets (SBUs) and Multiple-Bit Upsets (MBUs) for both L1 and L2 memories. The results show that the SBUs on L1 and L2 memories are the most common events observed in the experiments, i.e., 76.4% and 65.3%, respectively. Contrarily, MBUs have a lower occurrence when compared to SBUs. For L1, 12.6% for 2 MBUs and 11.1% for 3 or more MBUs. For L2, 21.3% for 2 MBUs and 13.4% for 3 or more MBUs.

We observe different MBU patterns for the two GAP8 levels of memory. In 0.9% of the cases for L1 memory, we observe large MBUs with 10 to 18 bits. In 1.1% of the cases for L2, the MBUs have 7 or 9 bits. For L1 and L2, 18 and 9 are the maximum number of MBUs observed, respectively. GAP8 uses a Tightly-Coupled Data Memory (TCDM) to increase performance and save energy, and, as observed in prior works, the MBUs can be spatially random for some memories types (i.e., caches) depending on the cell organization and the access type [27]. A single particle can, in fact, lead to corruption on many bits of a memory array.

While less frequent, MBUs are much harder to be masked on computation and can increase the fault propagation probability by at least 50% [29]. Additionally, it has been shown that the patterns of the MBUs can range in the number of bits flipped [27], [30]. That is, the same particle may modify the value of an entire word (16 bits) in the memory. Based on the high cross-section and high criticality observed on the GAP8's memory reliability analysis, we expect memory errors significantly impact the overall code cross-section. In the next section, we show how the code's cross-section changes based on the code's memory usage.

B. CNN operations cross-section

Figure 4 depicts the experimentally measured SDC and DUE cross sections. The y-axis shows the cross-section, and the x-axis shows each combination of operations (Convolution, Linear, & Maxpool) by execution type (sequential & parallel).

For all CNN operations, the SDC cross-sections of sequential algorithms are similar to parallel ones. The SDC rate differences of sequential and parallel versions are 9.2% and 20.1% for Convolution and Maxpool, respectively. Note that, even if the different implementations of the same algorithms have a significant performance difference (Table I), the memory error rate significantly contributes to the error rate. The highest difference comes from the Linear operation, in which

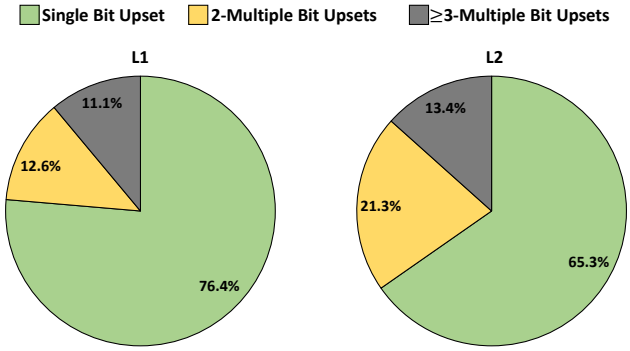

Fig. 3: Distribution of observed Single-Bit Upsets (SBUs) and Multiple-Bit Upsets (MBUs) on L1 and L2 microbenchmarks.

the parallel version is 35% higher than the sequential one. The parallel version of the linear Layer is performed entirely by vector instructions, stressing the functional units of GAP8, which leads to an increasing error rate for those units.

The SDC rate results show that the higher the memory resource usage, the higher the SDC rate. As the Convolutional layer is the most resource-demanding, it has the highest error rate. The same happens for Linear and Maxpool operations. Maxpool layers also can mask most of the faults, as 3/4 of the values are discarded as expected (details Section III).

On the contrary, the DUE cross-sections are much lower than the SDC ones. The SDC cross-section is on average 2.1×10^{-9}, while the DUE cross-section is on average 3.4×10^{-10}. DUEs are caused mainly by events unrelated to arithmetic calculations, e.g., illegal instructions, hangs due to a fault-induced infinity loop, fault-induced deadlocks, incorrect addresses for jump and branch instructions, and illegal memory accesses. Not surprisingly, the highest DUE cross section is the Maxpool operations (5.1×10^{-10}), which consists of multiple *max* instructions on memory blocks of the input.

Finally, the presented cross-sections and evaluations are essential to understand how faults impact a CNN layer on a RISC-V-based processor. Knowing the cross-section of each operation on a CNN allows us to calculate the overall cross-section on a CNN once the probability of the fault propagating is available, i.e., the Architectural/Program Vulnerability Factor. As the GAP8 is based on an open-source SoC (PULP RISC-V), it allows researchers to perform fault injection at different levels of abstraction, such as system and microarchitectural levels. In future works, we will demonstrate the impact of each CNN operation based on the fault propagation from fault injection and apply it to realistic CNN models.

V. Conclusions

We have discussed the cross-section of CNNs core operations executing on an ultra-low-power RISC-V SoC exposed to a beam of neutrons. For each CNN operation, we evaluated the sequential and parallel versions of the algorithms. Additionally, to investigate the impact of unprotected memories on the code's reliability, the error rate of the main memories of GAP8 RISC-V is evaluated. Although parallel and sequential versions of the CNNs operations have remarkable differences

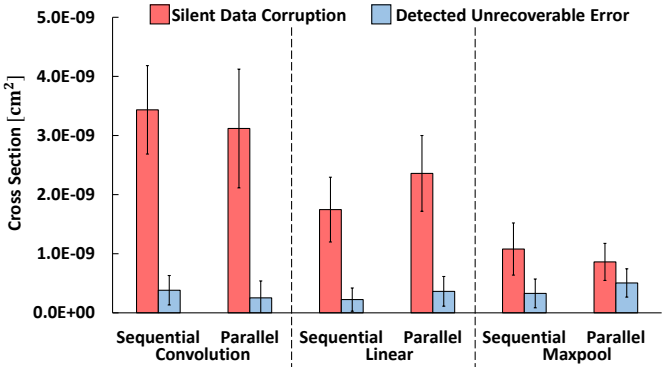

Fig. 4: Experimentally measured Silent Data Corruption (SDC) and Detectable Unrecoverable Errors (DUE) Cross Sections.

in performance, the error rate is still majorly driven by the error rate of the memories. We plan to deeply study fault propagation through CNN operations and realistic CNN models by performing system and microarchitectural level fault injection.

Acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 899546 with the support of the Brittany Region and partially funded by ANR FASY (ANR-21-CE25-0008-01) and ANR RETRUSTING (ANR-21-CE24-0015-02). ChipIR provided neutron beam time (DOI 10.5286/ISIS.E.RB2220502). We acknowledge the researchers that helped with neutron experiments, Dr. Christopher Frost, Maria Kastriotou and Dr. Carlo Cazzaniga.

References

[1] L. Lu, M. Zhang, and D. He, "Design and implementation of a smart home system based on the risc-v processor," in *2020 IEEE 2nd International Conference on Civil Aviation Safety and Information Technology (ICCASIT*, 2020, pp. 300–304.

[2] F. Ficarelli, A. Bartolini, E. Parisi, F. Beneventi, F. Barchi, D. Gregori, F. Magugliani, M. Cicala, C. Gianfreda, D. Cesarini, A. Acquaviva, and L. Benini, "Meet monte cimone: Exploring risc-v high performance compute clusters," in *Proceedings of the 19th ACM International Conference on Computing Frontiers*, ser. CF '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 207–208. [Online]. Available: https://doi.org/10.1145/3528416.3530869

[3] D. A. Santos, L. M. Luza, C. A. Zeferino, L. Dilillo, and D. R. Melo, "A low-cost fault-tolerant risc-v processor for space systems," in *2020 15th Design Technology of Integrated Systems in Nanoscale Era (DTIS)*, 2020, pp. 1–5.

[4] A. Ruospo, R. Cantoro, E. Sanchez, P. D. Schiavone, A. Garofalo, and L. Benini, "On-line testing for autonomous systems driven by risc-v processor design verification," in *2019 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, 2019, pp. 1–6.

[5] E. Flamand, D. Rossi, F. Conti, I. Loi, A. Pullini, F. Rotenberg, and L. Benini, "Gap-8: A risc-v soc for ai at the edge of the iot," in *2018 IEEE 29th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, 2018, pp. 1–4.

[6] A. Garofalo, M. Rusci, F. Conti, D. Rossi, and L. Benini, "Pulp-nn: accelerating quantized neural networks on parallel ultra-low-power risc-v processors," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 378, no. 2164, p. 20190155, 2020. [Online]. Available: https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2019.0155

[7] J. C. Laprie, "Dependable computing and fault tolerance : Concepts and terminology," in *Fault-Tolerant Computing, 1995, Highlights from Twenty-Five Years., Twenty-Fifth International Symposium on*, Jun 1995, pp. 2–.

[8] M. Nicolaidis, "Time redundancy based soft-error tolerance to rescue nanometer technologies," in *VLSI Test Symposium, 1999. Proceedings. 17th IEEE*, 1999, pp. 86–94.

[9] R. Baumann, "Soft errors in advanced computer systems," *2005 IEEE Design Test of Computers*, 2005.

[10] G. P. Dávila, D. Oliveira, P. Navaux, and P. Rech, "Identifying the most reliable collaborative workload distribution in heterogeneous devices," in *2019 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2019, pp. 1325–1330.

[11] D. A. G. Goncalves de Oliveira, L. L. Pilla, T. Santini, and P. Rech, "Evaluation and mitigation of radiation-induced soft errors in graphics processing units," *IEEE Transactions on Computers*, vol. 65, no. 3, pp. 791–804, 2016.

[12] J. M. Badia, G. Leon, J. A. Belloch, M. Garcia-Valderas, A. Lindoso, and L. Entrena, "Comparison of parallel implementation strategies in gpu-accelerated system-on-chip under proton irradiation," *IEEE Transactions on Nuclear Science*, pp. 1–1, 2021.

[13] K. Ito, Y. Zhang, H. Itsuji, T. Uezono, T. Toba, and M. Hashimoto, "Analyzing due errors on gpus with neutron irradiation test and fault injection to control flow," *IEEE Transactions on Nuclear Science*, vol. 68, no. 8, pp. 1668–1674, 2021.

[14] H. Quinn, P. Graham, J. Krone, M. Caffrey, and S. Rezgui, "Radiation-induced multi-bit upsets in sram-based fpgas," *IEEE Transactions on Nuclear Science*, vol. 52, no. 6, pp. 2455–2461, 2005.

[15] F. M. Lins, L. A. Tambara, F. L. Kastensmidt, and P. Rech, "Register file criticality and compiler optimization effects on embedded microprocessor reliability," *IEEE Transactions on Nuclear Science*, vol. 64, no. 8, pp. 2179–2187, 2017.

[16] R. L. Rech Junior, S. Malde, C. Cazzaniga, M. Kastriotou, M. Letiche, C. Frost, and P. Rech, "High energy and thermal neutron sensitivity of google tensor processing units," *IEEE Transactions on Nuclear Science*, vol. 69, no. 3, pp. 567–575, 2022.

[17] A. E. Wilson and M. Wirthlin, "Neutron radiation testing of fault tolerant risc-v soft processor on xilinx sram-based fpgas," in *2019 IEEE Space Computing Conference (SCC)*, 2019, pp. 25–32.

[18] A. B. de Oliveira, L. A. Tambara, F. Benevenuti, L. A. C. Benites, N. Added, V. A. P. Aguiar, N. H. Medina, M. A. G. Silveira, and F. L. Kastensmidt, "Evaluating soft core risc-v processor in sram-based fpga under radiation effects," *IEEE Transactions on Nuclear Science*, vol. 67, no. 7, pp. 1503–1510, 2020.

[19] D. A. Santos, L. M. Luza, M. Kastriotou, C. Cazzaniga, C. A. Zeferino, D. R. Melo, and L. Dilillo, "Characterization of a risc-v system-on-chip under neutron radiation," in *2021 16th International Conference on Design Technology of Integrated Systems in Nanoscale Era (DTIS)*, 2021, pp. 1–6.

[20] F. F. Dos Santos, A. Kritikakou, and O. Sentieys, "Experimental evaluation of neutron-induced errors on a multicore risc-v platform," in *2022 IEEE 28th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, 2022, pp. 1–7.

[21] A. E. Wilson, M. Wirthlin, and N. G. Baker, "Neutron radiation testing of risc-v tmr soft processors on sram-based fpgas," *IEEE Transactions on Nuclear Science*, pp. 1–1, 2023.

[22] G. Li, S. K. S. Hari, M. Sullivan, T. Tsai, K. Pattabiraman, J. Emer, and S. W. Keckler, "Understanding error propagation in deep learning neural network (dnn) accelerators and applications," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '17. New York, NY, USA: Association for Computing Machinery, 2017. [Online]. Available: https://doi.org/10.1145/3126908.3126964

[23] F. F. d. Santos, P. F. Pimenta, C. Lunardi, L. Draghetti, L. Carro, D. Kaeli, and P. Rech, "Analyzing and increasing the reliability of convolutional neural networks on gpus," *IEEE Transactions on Reliability*, vol. 68, no. 2, pp. 663–677, 2019.

[24] F. Libano, P. Rech, B. Neuman, J. Leavitt, M. Wirthlin, and J. Brunhaver, "How reduced data precision and degree of parallelism impact the reliability of convolutional neural networks on fpgas," *IEEE Transactions on Nuclear Science*, vol. 68, no. 5, pp. 865–872, 2021.

[25] JEDEC, "Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices," JEDEC Standard, Tech. Rep. JESD89A, 2006.

[26] C. Cazzaniga and C. D. Frost, "Progress of the scientific commissioning of a fast neutron beamline for chip irradiation," *Journal of Physics:*

Conference Series, vol. 1021, p. 012037, may 2018. [Online]. Available: https://doi.org/10.1088/1742-6596/1021/1/012037

[27] A. Dixit and A. Wood, "The impact of new technology on soft error rates," in *2011 International Reliability Physics Symposium*, 2011, pp. 5B.4.1–5B.4.7.

[28] Xilinx, "Device reliability report second half 2021 (ug116)," June 2021. [Online]. Available: https://www.xilinx.com/content/dam/xilinx/support/documents/user_guides/ug116.pdf

[29] A. Chatzidimitriou, G. Papadimitriou, C. Gavanas, G. Katsoridas, and D. Gizopoulos, "Multi-bit upsets vulnerability analysis of modern microprocessors," in *2019 IEEE International Symposium on Workload Characterization (IISWC)*, 2019, pp. 119–130.

[30] J. Suh, M. Annavaram, and M. Dubois, "Macau: A markov model for reliability evaluations of caches under single-bit and multi-bit upsets," in *IEEE International Symposium on High-Performance Comp Architecture*, 2012, pp. 1–12.