Analyzing the Reliability of Stream Sparse Matrix-Vector Multiplication Accelerators: A High-Level Approach

Analyzing the Reliability of Stream Sparse Matrix-Vector Multiplication Accelerators: A High-Level Approach

María L. Pinto-Salamanca*‡, Josie E. Rodriguez Condia†, José A. Hidalgo-López‡, Wilson J. Pérez-Holguín*
*Universidad Pedagógica y Tecnológica de Colombia (UPTC) - GIRA Research Group
{ marialuisa.pinto, wilson.perez}@uptc.edu.co
†Politecnico di Torino - Department of Control and Computer Engineering (DAUIN)
josie.rodriguez@polito.it
‡ Universidad de Málaga - Departamento de Electrónica
{mlpintos, jahidalgo}@uma.es

***Abstract*—[1] Today, advances in scientific and embedded computing and the incredible proliferation of machine learning algorithms take advantage of specialized hardware accelerators to provide exceptional performance and good accuracy. In some safety-critical applications like autonomous robotics, healthcare, and automotive, crucial mathematical operations such as convolutions are often efficiently mapped in hardware as Matrix-Vector*(MxV)* and Matrix-Matrix *(MxM)* multiplications. Moreover, sophisticated sparsity algorithms aim to improve performance and power consumption. Unfortunately, technology scaling trends may increase the proliferation of faults on hardware during the in-field operation, so there is a rising interest in the reliability assessment and analysis of sparsity-based accelerators. This work evaluates the impact of soft errors (bit-flips) on sparse-matrix dense-vector multiplication *(SpMV)* cores for safety-critical tactile sensing applications by resorting to a High-Level Synthesis (HLS) strategy. The experiments are performed on an open-source streaming SpMV core using the Compressed Sparse Row *(CSR)* format when processing characteristic medium-size sparse matrices (100x100 and 494x494). Our results indicate that data-path pipeline registers in the *(SpMV)* core are resilient to transient faults ($< 1.3\%$ of observed corruption effects), while large magnitude errors can be associated with the type of sparse matrix describing the system (e.g., number of non-zero values) and the type and features of the input vector sensor.**

***Index Terms*—Sparse Matrix Cores, Stream architecture, Reliability Assessment, Transient faults, Tactile sensing.**

I. Introduction

Electronic skin systems (*e-Skin*) integrate multimodal haptic capabilities (i.e., sensor arrays and computer vision) that, in combination with specialized devices, provide human-computer interaction and drive the operation of specialized tasks (such as exoskeletons, collaborative robots, surgical assistance robots, and augmented reality) [1]. These systems are characterized by high data density and high-resolution (originated by large amounts of tactile sensor arrays), enhanced data processing functions (integrating artificial intelligence and machine learning), operational performance, power consumption, and complex operating algorithms, mostly operating under real-time constraints (e.g., <1 ms). Thus, tactile sensing applications involve sophisticated and specialized hardware to support its deployment on edge devices combined with robust high-performance computing systems [2].

In robotics, most physical dynamics of tactile systems are modeled using large matrices representing the main operational characteristics of the sensors, actuators, mechanics, and devices involved in a complete system [3]. To address real-time data processing, robotic systems use *Tactile Controllers* to support the configuration and demanding data processing of sensing functions. These controllers use specialized hardware platforms (e.g., FPGAs, DSPs, or GPUs) to implement effective data transmission architectures that focus on matrix operations, such as *General Matrix-Multiply* (GEMM) algorithms and efficient compression, which are massively applied in various domains, including deep neural networks [4]. In this regard, architectures of sparse matrix multiplication algorithms can be extensively used (in up to 70% of all operations [5]) to optimize and speed up the processing time while efficiently using the hardware resources. The challenges they involve explain the growing interest of the scientific community in evaluating the reliability of the hardware underlying smart tactile sensing to identify vulnerabilities and support design improvements in this field.

In the literature, several works identified reliability threats and challenges in implementing tactile and *e-Skin* systems. In [6], the authors analyzed security issues in surgical robots and showed that corruption in communications could crash the system. Their results show that software vulnerabilities are real and could stop critical operations. Unfortunately, the analysis was limited to the operative system's vulnerabilities in robots, neglecting their underlying processing hardware. In [7], the authors highlighted the need for reliable and robust robot ecosystems focusing on the interaction among subsystems. On the hardware side, some works [8], [9] focused on analyzing

[1]This work has been partially supported by the Spanish Government under Grant PID2021-1250910B-I00, and by the National Resilience and Recovery Plan (PNRR) through the National Center for HPC, Big Data and Quantum Computing.

and detecting fault effects on accelerators performing matrix operations showed several vulnerabilities in the underlying architecture. Unfortunately, to the best of our knowledge, most works neglected analyzing the effects of failures in data flow architectures on the operation of this class of systems.

In this work, for the first time, we analyze and evaluate the vulnerability to transient faults in sparse matrix hardware accelerators when implementing force estimation in safety-critical tactile sensing applications. We employ a high-level approach based on *High-Level Synthesis* (HLS) to balance evaluation performance and error impact accuracy efficiently [10]. In particular, we include one fault injection infrastructure to accelerate hardware evaluation while providing appropriate accuracy.

For the experiments, we use an open-source sparse matrix-vector accelerator (SpMV) implementing a *Modified Compressed Sparse Row* (MCSR) algorithm on a streaming dataflow engine [11]. The evaluation focused on corrupting the most significant bits of the core through transient faults (*bit-flips*) when running four representative tests for industrial robots. Our results show that pipeline registers are remarkably resilient to transient faults, with less than 1.3% of the corruption effects observed. However, we note that large-magnitude errors might arise in the application's output due to the amount of non-zero values on the characteristic sparse matrix and the type of input vector (sensor) representing force estimation effects.

II. Sparse Matrix Multiplication on Modern Architectures

Matrix-Vector *(MxV)* and Matrix-Matrix *(MxM)* multiplication are fundamental primitives to support the implementation of optimized algorithms in several scientific domains. Efficient matrix algebraic operations are developed considering mixed density and sparse optimizations *(mostly zeros)* to reduce the overall amount of operations (i.e., neglecting multiplications with operands equal or near zero). In particular, sparse-matrix dense-vector SpMV operations are defined as $y = Ax$, with a sparse matrix(A) and a dense vector (x). Due to the sparse nature of A, the multiplication can be efficiently compressed using coding and data-compression strategies that only store the non-zero values of A and their location indices in the matrix. Those coding and data-compression strategies include the *Compressed Sparse Row* (CSR) and *Compressed Spare Column* (CSC), which only store the row-column and column-row indices of non-zero values in A, respectively.

The sparse algorithms are efficiently adapted and deployed on most hardware architectures, including streaming data flow and cascade, sparse TCU-cores, GPUs, vector accelerators (e.g., IPUs), and multi-threading processors. Custom streaming data flow and cascade architectures in online systems are crucial due to the minimal execution latency. These streaming architectures comprise high-bandwidth memories, one or more fast input memory buffers, one or several execution *kernels/processing elements* 'PEs' (e.g., *Multiply and Add* or MAC cores) organized in parallel on in cascade (i.e., pipeline) to process large amounts of data efficiently.

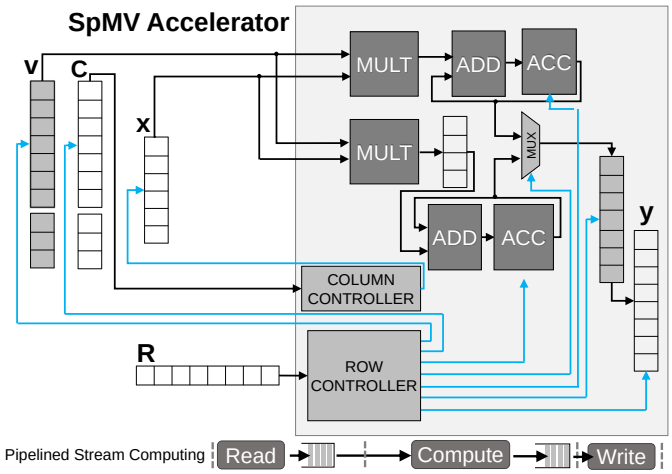

Fig. 1. General scheme of the proposed SpMV Accelerator.

III. A High-Level Approach to Assess the Reliability of Sparse Matrix-Vector Cores

Our approach exploits an HLS abstraction to include hardware infrastructures and evaluate soft errors during the core execution, as from the below three steps:

A. SpMV Characterization

This step characterizes the structures of a SpMV accelerator by identifying its functional operations and determining their pipeline registers in its execution path. The accelerator is based on a *Streaming Dataflow Engine* [11] and described in HLS that suppress its internal structural details, so we resort to its RTL description (*after synthesis*) to characterize its internal units (see Figure 1). The streaming core processes and passes an input data stream (from a sensor array) to the x. Moreover, one constant sparse matrix A (modeling the system) is compressed using an MCSR format and stored in three vectors: v, C, and R that contain the non-zero values (Nnz), the positions of the columns for Nnz of A (i.e., the x-components to be multiplied), and the number of multiplications per each row in A (MPR), respectively. Vector **R** also controls the number of readings of **v** and **C**.

B. Fault injection infrastructure

This step instruments the SpMV accelerator with a generic hardware fault injector structure *(saboteur)* targeting the pipeline registers of the kernel core in SpMV. We use an HLS description to dynamically place the *Saboteur* that corrupts the values before each register by mutating an input value to build the fault effect (i.e., *permanent* or *transient*). Figure 2 illustrates the general scheme of the saboteur-based (SB) fault injector. This structure can be dynamically included in the SpMV accelerator to address any possible sub-structure (e.g., output registers in the multipliers). In particular. the proposed SB allows the fine-grain control of a corrupting bit *(k-th)* inside a targeted register. Moreover, for transient faults *(bit-flips)*,

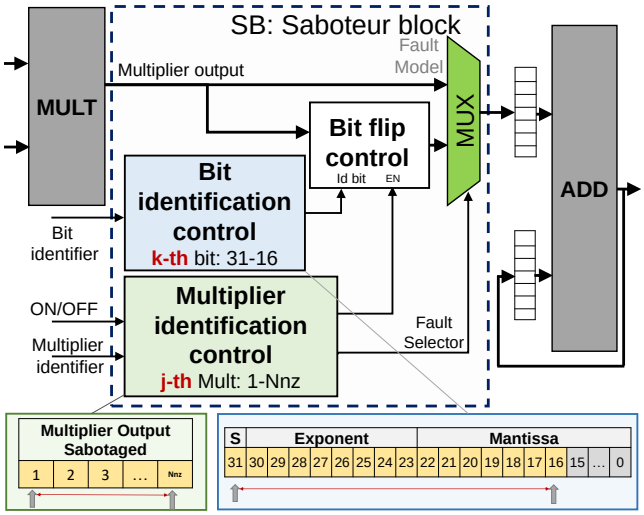

Fig. 2. A general scheme of the proposed fault injection infrastructure.

one identification control module enables the corruption on a specific operative cycle *(j-th)* of the SpMV accelerator.

C. Fault Evaluation and Error Classification

We evaluate the impact of transient faults through fault injection campaigns on the streaming execution of the SpMV core to typical industrial robot workloads. Each workload is evaluated considering several x input vectors representing online information from tactile sensor arrays to simulate non-continuous contacts (x_{nc}) and continuous contact (x_c) in which an incremental normal force is exerted. It is worth noting that each fault injection only generates a transient fault.

The impact on the result of the workloads from faults in the accelerator is classified as *Silent Data Corruption* (SDC) when the output values are corrupted, and a mismatch concerning the reference value is observed. Similarly, fault effects are classified as *Detected Unrecoverable Error* (DUEs) when the operation of the spare matrix core crashes, stops, or a Not-a-Number (NaN) condition is triggered. Finally, a fault effect is classified as *Masked* when the output values do not present any corruption effects concerning the reference operation.

IV. Experimental Results

Our experimental setup for the SpMV accelerator was configured to use single-precision (32-bit) floating-point data and input buffers of 100 or 494 elements, corresponding to two matrix-vector configurations (100×100 or 494×494). We include a fault injection infrastructure (saboteur) that only evaluates the most significant part (16 bits) of the pipeline registers on the multiplier outputs in the SpMV accelerator's MAC since fault effects on the less significant part mainly produce error effects near zero.

We employ six input vectors representing the non-continuous input force contact events (x_{nc}) and continuous (and rising force) contact (x_c) sensors scenarios. In the first scenario ($x = x_{nc}$), five vectors comprise random values ranging from 0.0 to 1.0, while in the second, one vector ($x = x_c$) swaps from 0.0 to the size of matrix A. More

TABLE I
Characteristics of the sparse matrix benchmarks.

Benchmark	Size	Nnz	TN
A1	100 ×100	708	11,328
A2	100 ×100	2,419	38,704
A3	494 ×494	1,666	26,656
A4	494 ×494	6,744	107,904

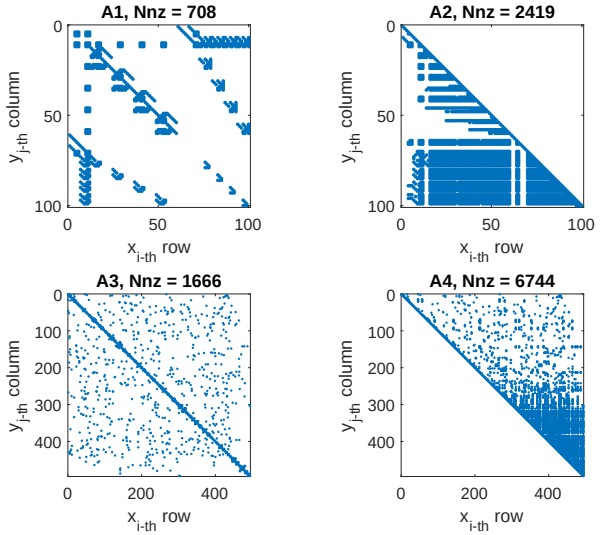

Fig. 3. Shape of the evaluated Sparse Matrix Benchmarks.

in detail, we use four typical industrial robots sparse matrix (A) benchmarks (depicted in Figure 3) from the *Sparse Matrix Collection* suite [12], see Table I. Since the sparse compression mechanism eliminates all operations with zero values, the number of non-zero (Nnz) elements is related to the number of evaluated faults per benchmark (TN). We applied 24 fault injection campaigns to evaluate the impact of 184,592 transient faults (*bit-flips*) corrupting the pipeline registers. Each evaluation considers one fault injected at a random time. All experiments were performed on the *Vitis™ HLS 2022* framework on a system with an Intel Dual-Core i7-4600U CPU at 2.10GHz and 8GB of RAM.

To determine the overall cost in hardware resources of the SB structure in the SpMV accelerator, we identify the number of logical elements after synthesis in a Zynq UltraScale+ MPSoC ZCU102 (xczu9eg-ffvb1156-2-e) FPGA. Particularly, the *MULT*, *ADD*, and *ACC* units are mapped using the DSP slices on the FPGA device. Our analysis shows that the SB infrastructure increases the hardware overhead by approximately 20% and 32% of the FF and LUT used, respectively, while the number of BRAM and DSP slices remains the same in both cases. Likewise, we determine the structural impact of transient faults in the SpMV accelerator by determining the fault rate for each benchmark.

Figure 4 depicts the fault rate as SDCs, DUEs, and masked effects, showing that transient faults in the data path of the accelerator can hardly corrupt operations and produce mismatches on the outputs (1.3% of SDCs), regardless of the size of A and the x input scenario. Moreover, faults do not caused NaN exceptions or hanging effects (i.e., DUEs) due

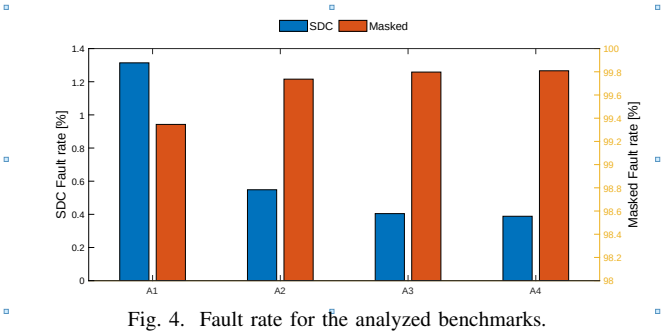

Fig. 4. Fault rate for the analyzed benchmarks.

to the operative ranges of the characteristic matrix (A) and each tactile contact model (x) in the application. The masking effects in our results (around 99.7% for all operative scenarios) indicate that streaming sparse matrix-vector cores are resilient to most transient faults arising in data-path registers, particularly in data-flow operations of force estimation for tactile sensing.

Analyzing the sparse matrix benchmarks, we observe that the A1 matrix is up to three times more vulnerable to corruption due to transient failures (around 1.3%) than other sparse benchmarks analyzed (from around 0.4% to 0 .55%), which may be due to a smaller variation in the number of multiplications per row and a smaller Nnz/A_{size} ratio.

An analysis of the SDC errors suggests a relation between the sparse matrix type and the generation of large magnitude impacts ($\sim 3.4 \times 10^{38}$) for both operational sensor scenarios (non-continuous x_{nc} and continuous x_c). The results of maximum relative error (see Figure 5) for each output (y), on the evaluated sparse matrix when considering both sensor scenarios (x_{nc} and x_c), and the number of corrupted multiplications per each row in A (MPR), show that non-continuous x_{nc} sensing inputs are prone to produce large-magnitude error effects on most output elements. In contrast, x_c continuous contact inputs slightly reduce the maximum error effects, which is consistent for all analyzed matrices.

V. Conclusions and Future Work

This work evaluated and characterized the vulnerability of pipeline registers in streaming sparse matrix multiplication cores to transient faults when performing tactile operations in critical robotic applications. We take advantage of a high-level synthesis approach to place saboteur-based fault injectors devoted to speeding up the system evaluation.

The results show that the pipeline logs are highly resilient to transient failures by up to 99.8%. In the same way, our results suggest that large-magnitude errors in tactile applications arise as a relation between the number of non-zero elements in the sparse matrix and the input sensor type. Continuous force sensors are more vulnerable to faults and prone to application corruption than non-continuous sensors.

We plan to extend our analysis to other architectures and propose hardening mechanisms by combining high-level synthesis with fine-grain verification techniques.

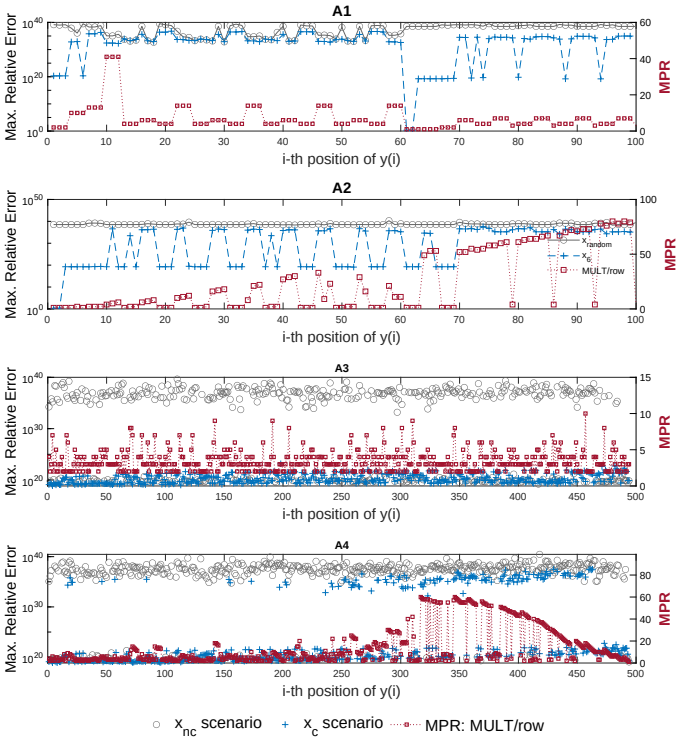

Fig. 5. Maximum relative error and occurrences of corrupted operations per evaluated benchmark.

References

[1] R. Dahiya et al., "Large-area soft e-skin: The challenges beyond sensor designs," Proc. of the IEEE, vol. 107, no. 10, pp. 2016–2033, 2019.

[2] P. Roberts et al., "Soft tactile sensing skins for robotics," Current Robotics Reports, vol. 2, pp. 343–354, 2021.

[3] S. M. Neuman et al., "Roboshape: Using topology patterns to scalably and flexibly deploy accelerators across robots," in Proc. of the 50th Annu. Int. Symp. on Computer Architecture (ISCA 23), 2023.

[4] S. Han et al., "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," arXiv preprint arXiv:1510.00149, 2015.

[5] M. L. Pinto-Salamanca et al., "An estimation of triaxial forces from normal stress tactile sensor arrays," Mechatronics, vol. 96, p. 103070, 2023.

[6] K. Chung et al., "Smart malware that uses leaked control data of robotic applications: The case of Raven-II surgical robots," in Int. Symp. on Research in Attacks, Intrusions and Defenses (RAID), 2019, pp. 337–351.

[7] M. Dohler, "The tactile internet iot, 5g and cloud on steroids," in 5G Radio Technology Seminar. Exploring Technical Challenges in the Emerging 5G Ecosystem, 2015, pp. 1–16.

[8] W. Lin et al., "Resource-aware online permanent fault detection mechanism for streaming convolution engine in edge ai accelerators," in IEEE Int. Conf. On Artificial Intelligence Testing (AITest), 2023, pp. 132–137.

[9] P. M. Basso et al., "Impact of tensor cores and mixed precision on the reliability of matrix multiplication in gpus," IEEE Trans. Nucl. Sci., vol. 67, no. 7, pp. 1560–1565, 2020.

[10] A. Orailoglu and R. Karri, "A design methodology for the high-level synthesis of fault-tolerant asics," in Workshop on VLSI Signal Processing, 1992, pp. 417–426.

[11] M. Hosseinabady and J. L. Nunez-Yanez, "A streaming dataflow engine for sparse matrix-vector multiplication using high-level synthesis," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 39, no. 6, pp. 1272–1285, 2020.

[12] S. P. Kolodziej et al., "The suitesparse matrix collection website interface," J. Open Source Softw., vol. 4, no. 35, p. 1244, 2019.