Evaluation of a robust system for hyperspectral data compression under neutron and proton effects

Luis Entrena, Antonio J. Sánchez, Mario García-Valderas, Yubal Barrios, Roberto Sarmiento, Almudena Lindoso

***Abstract*—The use of hyperspectral sensors in space missions is steadily increasing. These sensors generate large volumes of data in the form of hyperspectral cubes (images captured in hundreds of different wavelengths) which require the adoption of data compression solutions onboard. In this work we evaluate error mitigation for CCSDS-123 hyperspectral data lossless compression IP. Realizing that a data compressor has to be used with some data interfaces and possibly a processor, the evaluation is performed for entire systems built around the SHyLoC CCSDS-123 IP core. To this purpose, we designed hardened systems that include redundant IP cores, input and output data stream interfaces, and processors. The hardened systems were implemented in a commercial SoC device and tested with atmospheric neutrons and protons. Experimental results are presented and discussed for two architectures using dual and triple IP core redundancy, respectively.**

***Index Terms*— Hyperspectral imaging, data compression, Single Event Upset, soft error, FPGA, MPSoC**

I. INTRODUCTION

MULTISPECTRAL and hyperspectral sensors are being embarked on a growing number of Earth Observation (EO) satellites [1]. This trend is reflected in their use not only in current space missions (e.g., the EMIT instrument employed in the International Space Station (ISS) [3] or on-board some satellites, such as EnMAP [4] or PACE [5]), but also in their inclusion in the roadmap of space agencies, such as the Copernicus Hyperspectral Imaging Mission for Environment (CHIME) supported by European Space Agency (ESA) [6].

These sensors produce large multidimensional datasets that contain information across a wide range of wavelengths [2]. Although its use provides several benefits to the scientific community thanks to the quantity of information acquired, this high data volume can be neither stored on-board nor transmitted in raw format (i.e., as it is collected). Thus, compression is mandatory to reduce on-board storage and to meet available data transmission bandwidth, preserving as much information as possible [7].

Some important aspects to consider when implementing hyperspectral image compression solutions for its use on space are the existing constraints not only in terms of hardware and storage resources, but also in terms of throughput to accomplish real-time processing demands.

The Consultative Committee for Space Data Systems (CCSDS) has issued several standards for satellite payload data compression that have been adopted by space agencies. Among them, the CCSDS 123.0-B-1 standard [8] focuses on lossless compression of multi- and hyperspectral images and it is based on the Fast Lossless (FL) compression algorithm. This standard includes two main stages: a predictor used to exploit the correlation among input samples, both in the spatial and the spectral domains; and an entropy coder employed to represent the prediction residuals (i.e., the predictor output) with the shortest possible number of bits.

Several hardware implementations of this standard are available in the state-of-the-art. Among them, we found the SHyLoC CCSDS-123 IP core, which is a technology-independent implementation of the CCSDS 123.0-B-1 standard that is part of IP core's portfolio from ESA for space missions [9]. In addition, it was designed to meet real-time requirements imposed by typical on-board imaging sensors.

The CCSDS-123 IP core can be implemented in a variety of technologies, and it is commonly implemented on SRAM-based FPGAs to reduce costs and development effort. Rad-hard and rad-tolerant FPGAs can be used for this purpose. For instance, a Virtex-5QV FPGA was used in [17] for a 3.3 Gbps optimized implementation. Nevertheless, the use of Commercial Off-The-Shelf (COTS) SRAM-based FPGAs is very attractive because of their higher availability and performance, and lower power consumption. SoC (System on Chip) devices that embed one or multiple processor cores can even support pre-processing and post-processing tasks on chip. However, these devices have significant sensitivity to Single

This work has been supported in part by project PID2022-138696OB-C21 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe" and by the European Union's 2020 research and innovation programme under grant agreement No 101008126, corresponding to the RADNEXT project.

Luis Entrena. Mario Garcia-Valderas and Almudena Lindoso are with the Dept. of Electronic Technology, Universidad Carlos III de Madrid, Avda. Universidad 30, E-28911 Leganes, Madrid, Spain (e-mail: entrena@ing.uc3m.es; mgvalder@ing.uc3m.es; alindoso@ing.uc3m.es)

Antonio J. Sánchez, Roberto Sarmiento and Yubal Barrios are with the Institute for Applied Microelectronics (IUMA), University of Las Palmas de Gran Canaria (ULPGC), Las Palmas de Gran Canaria, Spain (e-mail: ajsanchez@iuma.ulpgc.es; roberto@iuma.ulpgc.es; ybarrios@iuma.ulpgc.es).

Event Effects (SEEs) and therefore it is necessary to apply error mitigation techniques.

The analysis of the vulnerabilities of FPGA implementations for hyperspectral data compression has been addressed in some recent works. In [10], fault injection campaigns were performed on the unhardened SHyLoC CCSDS-123 IP core and two hardened versions using Dual Modular Redundancy (DMR) and Triple Modular Redundancy (TMR), respectively. However, this analysis was based on injecting faults in the configuration memory of the FPGA. Grignani et al. [16] presented a low-cost implementation of a fault-tolerant CCSDS 123 compressor using TMR for the control units and Hamming Error Correcting Code (ECC) for internal memories and registers. This low-cost fault-tolerant implementation has a reduced area overhead, but it also reduces performance by about 30%. Error mitigation was validated by fault injection in the protected memory elements of the compressor using a simulation approach, and neglecting other sensitive elements of the Zynq-7020 SoC device used for the implementation.

In [11], the effectiveness of using TMR for soft error mitigation was evaluated for a closely related core, the SHyLoC CCSDS-121 IP core. Tests were performed at CERN using heavy ions. TMR showed a moderate improvement in robustness with respect to the unhardened implementation. However, the results showed only a 40% improvement in the Mean Time To Failure (MTTF) due to the lack of protection of some components of the design, such as the Direct Memory Access (DMA) interface, the Advanced eXtensible Interface (AXI) infrastructure or the voting logic. In fact, this analysis revealed that focusing on the reliability of the payload data compressor alone is not generally sufficient. Whatever the application, the sensitivity of the compressor's input and output data stream interfaces, and possibly the downstream processor, must be considered, because they will be present in any system using the compressor and may have a significant impact in the integrity of the compression tasks.

In this work, we consider a robust system built around the SHyLoC CCSDS-123 IP core and perform radiation experiments intended to evaluate and validate error mitigation on the entire system. In contrast to previous works, which only protect the hyperspectral compressor, resulting in small reliability improvement [16], or use fault injection for evaluation [11], which can be conveniently restricted to the protected elements, our system includes redundant interfaces and processors, along with redundant implementations of the IP core, and evaluates the soft error reliability of the system under proton and neutron radiation. The main contribution of this work is to demonstrate that a robust implementation can be obtained and validated under radiation using a system-level approach. To this purpose, the compressor is implemented on a commercial Zynq-7000 All-Programmable System-on-Chip (APSoC) device that includes programmable logic along with a dual core processor. Each processor sends hyperspectral images to a compressor core implemented on Programmable Logic (PL) and collects the compressed data. The processors work in a Macrosynchronized Lockstep (MSLS) mode, being able to store valid checkpoint states and roll back to a previous checkpoint in case of error [12].

Two architectures are evaluated based on DMR and TMR of the hyperspectral data compression IP core, respectively. Radiation tests were performed with atmospheric neutrons at ChipIR (UK) and low energy protons at Centro Nacional de Aceleradores (CNA) in Seville, Spain. The goal was to evaluate the impact of soft errors in the result, correcting errors whenever possible and otherwise carrying out error recovery actions.

The remaining of this paper is as follows. Section II introduces related work on robust system implementations with FPGAs and SoCs. Section III presents the SHyLoC CCSDS-123 IP core. Section IV describes the hardened system architectures that are evaluated. Section V shows the experimental results. Finally, section VI presents the conclusions of this work.

II. Related Work

Radiation effects in SRAM-based FPGAs have been widely studied [21], [22] and effective error mitigation techniques have been proposed [23], [24]. In [24], the authors evaluate several mitigation techniques and compare them to TMR in both area cost and fault tolerance. This study confirmed the relevance of TMR as a SEU mitigation technique in SRAM-based FPGAs and concluded that no other technique has been demonstrated that produces lower-cost reliability.

There are also many works that evaluate radiation effects on COTS microprocessors [25], [26], and error mitigation approaches based on hardware redundancy [27], [28] or software redundancy [29], [30]. CNES (Centre National d'Études Spatiales) has proposed and patented two fault-tolerant techniques: Dual Multiplexed in Time (DMT) and Dual Duplex Tolerant to Transients (DT2). DMT uses time redundancy and DT2 uses spatial redundancy [27].

The increasing availability of multicore devices provides support for the implementation of spatial redundancy approaches. An effective solution is to use a device that has at least two cores able to work in Dual-Core Lockstep (DCLS) mode, such as Cortex-R5 from ARM [31] or Texas Hercules microprocessor [32]. These processors include hardware mechanisms to microsynchronize two cores and execute simultaneously the same instruction. Lately, some proposed works implement lockstep techniques with processors that lack this capability in their architecture. In [33], lockstep is applied to ARM Cortex A9 processor and validated with heavy ions, achieving a reduction of cross-section of one order of magnitude. A hybrid technique is presented in [34] and validated with proton irradiation. The proposed technique combines thread replication with an external IP to observe the correct behaviour of a Cortex A9 processor. In a previous work, we proposed a Macrosynchronized Lockstep (MSLS) approach to harden multicore processors [15]. The case study was a Cortex-A9 processor validated with protons, which accomplished a reduction of 2 orders of magnitude in cross-section.

A TCLS (Triple Core Lockstep) architecture for a Cortex-R5 ARM microprocessor was presented in [36]. The proposed technique is able to perform system recovery in a fast way (microseconds), achieving with it a decrease of downtime of the system. A TCLS scheme with a quad core ARM Cortex A53 as case study was presented in [37]. In this case, the system can perform roll-back and roll-forward, and experimental results showed a very high effectiveness in error detection and system recovery. TCLS has also been recently applied to a RISC-V architecture in the Trikarenos chip [38].

Multiprocessor Systems-on-Chip (MPSoCs), which include multiple processor cores and programmble logic in the same device, are very attractive to implement systems with tight size, weight and power consumption constraints. These devices need to use appropriate techniques to protect each part of the MPSoC, and make sure the parts interact in a robust manner. An analysis of the impact of radiation-induced failures in Programmable SoCs is provided in [38], considering several memory organizations, communication schemes, and computing modes. This work is extended in [39] with an analysis for hardware and software co-designs. However, these works focus on single-core architectures. Recently, we proposed an effective approach that implements modular redundancy in the programmable logic and Macrosynchronized Lockstep (MSLS) in the microprocessor cores [12].

With respect to hyperspectral data compression, reported implementations are either based on rad-hard FPGA [17], with no analysis of reliability and system-level repercussions, or are only partially protected [11], [16]. In this work, we focus on a SoC implementation, using a COTS device, and validate error mitigation on the entire system.

III. Description of the SHyLoC CCSDS-123 IP

The SHyLoC CCSDS-123 IP core is a hardware implementation of the CCSDS 123.0-B-1 compression standard [8], which describes a 3D predictive lossless compressor for multi- and hyperspectral data. It includes both the 3D predictor and the sample-adaptive entropy coder (Fig. 1), which is one of the encoding approaches allowed by the CCSDS 123.0-B-1 standard, in addition to the block-adaptive encoder.

The 3D predictor exploits the correlation in the spatial and/or spectral dimensions. With that aim, each input sample is estimated using a vicinity of previously processed samples. The local sum mode, prediction mode and number of bands used for prediction determine which neighbor samples are included in the prediction computation. Input samples are arranged in an array of FIFOs in such a way that the spatial neighbors can be easily located for prediction, from which local sums and local differences are derived. Similarly, local differences from previous bands are stored in FIFOs to be used in the prediction of future samples. To compute predicted samples, these local differences are weighted by internal weight vectors, which are constantly updated with the prediction results and aim to capture the statistical behavior of the input image. Prediction residuals result from comparing predicted and original samples, which are then mapped and encoded.

There are 5 different architectures of the SHyLoC CCSDS-123 IP 3D predictor adapted to the most common sample arrangement formats in multi- and hyperspectral sensors: Band-Interleaved by Pixel (BIP), Band-Interleaved by Line (BIL) and Band Sequential (BSQ), including 2 variants with support for external memory: BIP-MEM and BIL-MEM. All of them are pipelined and optimized for the dominant data dependencies, which may be different among formats.

The sample-adaptive encoder encodes mapped prediction residuals individually into variable-length codewords using the Rice coding [20], with a tunable parameter which is determined from some adaptive code selection statistics, constantly updated. This encoder is implemented in a pipelined fashion, with slight differences in architecture and control for the different sample arrangement formats supported. Nevertheless, the encoder achieves a performance of 1 sample per clock cycle in all sample arrangement formats.

Regarding internal complexity, the 3D predictor clearly overcomes the sample-adaptive encoder since it performs more complex arithmetic operations (e.g., dot products and multiplications) and stores more intermediate information. For this reason, it is the most critical module in case radiation effects take place. Regarding error mitigation techniques, the CCSDS-123 IP offers the possibility of enabling Error Detection And Correction (EDAC) in the internal memories by setting a specific configuration option. In case that an error is detected by the EDAC mechanism, the IP Error signal is asserted.

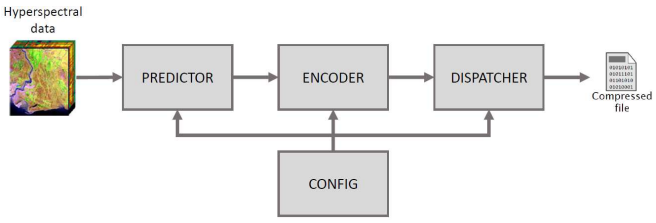

Fig. 1. SHyLoC CCSDS123-IP internal stages

Although the CCSDS-123 IP supports different data arrangements to be compatible with more employed hyperspectral sensors, the BIP architecture is selected for this study since it is the only option that achieves a maximum theoretical throughput of one sample per clock cycle, provided that the input image is large enough to fill the predictor pipeline.

In addition to the predictor and sample-adaptive encoder modules, the CCSDS-123 IP includes additional logic for control and configuration purposes, as shown in Fig. 1. The *config* module receives the user-defined configuration parameters, and validates and disseminates them to the main processing stages, while the *dispatcher* controls the output bitstream, concatenating the CCSDS 123 compression header (generated by an independent unit within the 3D predictor) to the bits coming from the encoder in fixed-length data words.

IV. Hardened System Design

If the target device is not rad-hard, error mitigation must be added by design. Hardware redundancy solutions based on

DMR or TMR are commonly used for FPGAs [13]. However, in the case of a data compressor it is necessary to consider the data interfaces and the processing subsystem as well. Therefore, we need to analyze the reliability of the compressor in the context of a system that has error mitigation capabilities.

The system architecture used in this work is based on [12]. It can be divided into two main parts:

- Processing System (PS): a multicore system that controls the computations performed and their correctness. The minimum number of processor cores that is required is two.
- Programmable Logic (PL), which contains several data compression IP modules to obtain the results using a redundant approach. The minimum number of IP modules that is required is two.

Redundant data interfaces are also used to connect the PS and the PL. The architecture of the data interfaces can vary depending on the number of processor cores and IP modules, and on the available interconnection resources.

As a case study, the system was implemented on a Zynq 7020 APSoC [14], which contains a hard-core PS with a dual core ARM Cortex A9, and PL to allocate several copies of the IP module. We have designed two different architectures, named Duplex architecture and Flexible TMR. These architectures are summarized in the diagrams of Fig. 2.

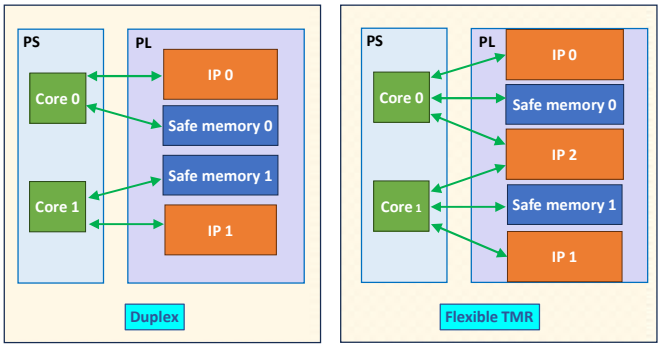

Fig. 2. Dual and Flexible TMR architectures.

In the Duplex architecture we use two processor cores and two data compression IP modules. Each PS core has a dedicated IP module in PL. Both PS cores execute the same application code in a macrosynchronized way [15], dividing the code in blocks and checking the correctness of the results after finishing the block execution. Errors are detected when a discrepancy is found in the results.

The PS system has rollback capabilities, being able to go back in the software application to the previous correctly executed program block. Rollback requires context storage to save checkpoint states. For this purpose, each core has an associated safe memory located in PL in our architecture. The amount of context information to be saved can be minimized by a good selection of the checkpoints. In our architecture, we use the safe memories just to store the contents of some registers and control variables that determine the state of the computation in a checkpoint.

In the Flexible TMR architecture we use two PS cores and three redundant IP modules. The two cores execute the same application code in a macrosynchronized way and have an associated safe memory for context storage, like in the Duplex architecture. The main difference in the Flexible TMR architecture is the third IP module implemented in PL, which provides additional information about the computations and can correct errors by voting.

With the use of three compression IP modules, we can detect the erroneous IP module or interface and repeat the computations in a more flexible way. Only one of the three IP modules, namely IP2, can connect to both PS cores in order to provide additional results, when needed. The other two IP modules are connected only to one core, in the same way it was done in the Duplex architecture. In case a discrepancy is found between the computations performed by the primary IP modules (0 and 1), the results from the third IP module (2) are considered. This result is sent to each core through a separate interface, so that they can be compared to detect possible errors in the interfaces.

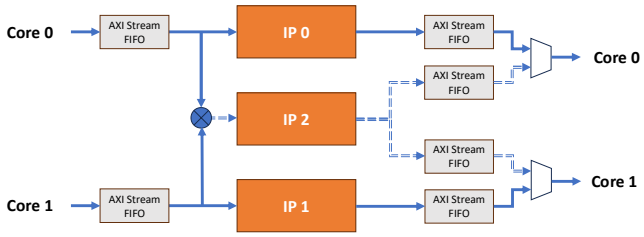

Fig. 3. Communication channels between the cores and the IPs.

The CCSDS-123 IP modules use the AXI4-Stream protocol to receive input data and transmit output data. The AXI4-Stream FIFO core [18] is used to interface the compressor IP modules and provide a memory mapped access to the input and output data. This IP core supports full duplex operation through independently configurable internal transmission and reception data FIFOs. Two separate instances of the AXI4-Stream FIFO core are used, each providing an input and an output channel to ensure the communications between each core and each IP are independent. In the Flexible TMR architecture, two additional AXI4-Stream FIFO cores are used to interface the outputs of IP2. Thus, in this case there are two data input channels and four data output channels, each with its own independent FIFO, as shown in Fig 3. The input to IP2 is taken from any of the input channels after checking they provide the same data.

The memory mapped side of the AXI4-Stream FIFO cores connect to the PS through AXI General Purpose (GP) interfaces. The Zynq 7020 APSoC has two AXI GP master interfaces between PS and PL, which are used for core 0 and core 1, respectively.

V. Experimental Results

A. Experimental setup

The proposed system-level solutions for the SHyLoC CCSDS-123 IP core were implemented on a Xilinx Zynq-7020 APSoC device that contains a dual-core ARM Cortex-A9

processor and SRAM-based Programmable Logic. We used a Zybo Z7 development board that mounts this device to carry out the experiments. The board includes 1GB of DRAM, but the code only occupies around 36kB for each core. Caches were disabled for simplicity. Each safe memory can store up to 80 kB of encoded data, but we used just a few words in our applications. Radiation experiments were performed with atmospheric neutrons at ChipIr (UK) and with protons 15 MeV protons at CNA (Spain).

In the experiments, we tested the Duplex and Flexible TMR architectures. The Duplex architecture includes two instances of the data compression IP core that are connected to the Processing System (PS) through separate AXI buses. Thus, each IP core is controlled by a dedicated processor. One of the processors, CPU0, is in charge of comparing the results and triggers a rollback when a discrepancy is found. The TMR architecture includes an additional instance of the IP core implemented in PL, that is used in case of discrepancy between the results provided by the two first instances.

The PS provides input data to compress to the compression cores implemented in the PL fabric. For these experiments we have used crops of real acquisitions from the Airbone Visible/Infrared Imaging Spectrometer (AVIRIS) instrument, with a fixed size of 55 x 55 pixels in the spatial dimensions and 224 bands.

In our experimental setup, the behavior of the system under test is controlled and observed by an external host that collects the results of the experiment through a serial communication channel. In case the device under test does not respond and cannot recover on its own, the external host performs a power cycle. For simplicity, scrubbing was not enabled, so errors in the configuration memory will generally be persistent and require a full reconfiguration to correct them.

B. Classification of errors

In the reports of the experiments, errors are classified into two main categories:

- *Detected error*: the system detects some malfunction, is aware of the error and can recover on its own.
- *Undetected error*: the system is no longer responsive and needs to be restarted by external means. The error is detected by the external host that observes the experiment, because the system is not able to recover on its own.

Detected errors are further divided into three groups, depending on the action needed to recover it:

- *Corrected*: When the system detects a data error, it triggers a rollback to a previous step. If this action solves the problem, producing correct results, we classify the error as Corrected.
- *Software reset*: After a predefined number of unsuccessful rollbacks, the processor resets the system.
- *Watchdog reset*: The processor may reset the system if it does not receive a response from the data compressors within a predefined time

Note that the category of Corrected errors exclusively refers to correction by rollback, i.e., by repeating part of the computations. In the Flexible TMR case, some errors may be corrected by voting. However, for the sake of completeness, in our experiments we always request the results from the three compressor instances and trigger a rollback in case there is any mismatch. In a real application, the system can provide a correct result even if one of the compressors fails, but the system will be at risk until the programmable logic is reconfigured and the wrong compressor is recovered.

Additional error correction may be obtained by using scrubbing. We did not implement scrubbing in the experiments because it may complicate the interpretation of the results, due to the high error rate that is produced in an accelerated radiation environment. In a real environment, scrubbing may eventually recover an IP given enough time, thus reducing the need to reset and reconfigure the programmable logic.

In the experiments, we have also observed a significant amount of communication errors, which result in wrong characters received through the serial data collection interface, and eventually in the collapse of the communication. In this case, the entire system is restarted but the error is not computed in the results, as it is not related to the goals of the experiments.

C. Radiation results

Tables I and II summarize the radiation results obtained for atmospheric neutrons and 15 MeV protons, respectively. In each table, errors are classified using the categories described in the previous subsection. The Tables also show the cross-section due to total errors and to undetected errors only.

TABLE I.
NEUTRON RADIATION RESULTS OF DUPLEX AND FLEXIBLE TMR ARCHITECTURES

	Category	Duplex	Flexible TMR
Errors	Detected	590 (99.5%)	1112 (99.6%)
	Undetected	3 (0.5%)	5 (0.4%)
	Total errors	593 (100.0%)	1117 (100.0%)
Detected errors	Corrected	118 (19.9%)	208 (18.6%)
	Software reset	259 (43.7%)	609 (54.5%)
	Watchdog reset	216 (36.4%)	300 (26.9%)
Cross-section (cm^2)	Total errors	1.09×10^{-8}	2.17×10^{-8}
	Undetected errors	5.53×10^{-11}	9.72×10^{-12}

TABLE II.
PROTON RADIATION RESULTS OF DUPLEX AND FLEXIBLE TMR ARCHITECTURES

	Category	Duplex	Flexible TMR
Errors	Detected	621 (98.7%)	1070 (99.1%)
	Undetected	8 (1.3%)	10 (0.9%)
	Total errors	629 (100.0%)	1080 (100.0%)
Detected errors	Corrected errors	89 (14.3%)	214 (20.0%)
	Software reset	316 (50.9%)	459 (42.9%)
	Watchdog reset	216 (34.8%)	397 (37.1%)
Cross-section (cm^2)	Total errors	1.43×10^{-10}	1.38×10^{-10}
	Undetected errors	1.82×10^{-12}	1.28×10^{-12}

The results show that the proposed system architecture is effective to mitigate soft errors caused by radiation in the hyperspectral data compression implementation. Comparing the cross-section due to the undetected errors with respect to the total errors, it is reduced by two orders of magnitude in both cases of protons and atmospheric neutrons. About 20% of the errors can be recovered by rollback, so they are not persistent. However, most of the errors require a reset of the system to recover. As the processors have a minimal workload, these errors are mostly due to the configuration memory of the FPGA, which may affect the data compressor IPs as well as the interfaces in a persistent manner.

In all experiments, the majority of errors require a software reset to be recovered. This means that the IPs complete the computation and produced a result within the expected time, but at least one of the results was incorrect and further attempts to recompute them also produced incorrect results. There is also a significant amount of errors that hang the system and require a watchdog reset, either because the PS is waiting for one of the IPs that is unable to complete the compression or because one of the processor cores no longer responds.

D. Distribution of errors in compressed datasets

The number of errors in a compressed dataset can vary widely. However, it has been experimentally observed that the distribution of the number of errors is not uniform. Fig. 4 shows a histogram obtained for a subset of the proton radiation experiments, in which we collected the number of words that differ from the expected result in each compressed dataset. The total size of the correct output dataset in this experiment is 509 words and we used a bin size of 40 words. The last bar corresponds to results that produce a larger dataset than expected.

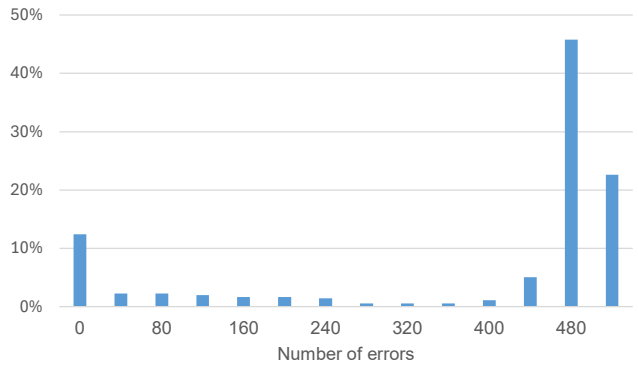

Fig. 4. Distribution of the number of incorrect words in compressed datasets under proton radiation

It can be clearly observed in Fig. 4 that most of the cases concentrate either around a few errors or a very large number of errors. The first case may be due to data errors or errors that affect the generation of the compression header. In fact, very few errors are single errors, which may have been directly produced by a Single Event Upset (SEU) in the output stream storage. However, in more than 40% of the cases, all or nearly all words were wrong. This behaviour points to errors in the predictor that may affect the entire compressed data. Finally, in some cases the compressor finished but produced a larger output. Interestingly, the majority of these cases fall in the category of corrected errors, i.e., they could be recovered by rollback. As errors in the configuration memory are persistent, we can assume that these errors are located in the interfaces or the processor cores inside the PS. As a conclusion, error detection and correction codes may be difficult to apply or ineffective, due to the large variety and multiplicity of errors that may appear in the compressed data streams.

E. Location of the errors

In some of the experiments, we collected information about the location of the error. In this case, we used the Flexible TMR architecture under neutron radiation and disabled rollback to simplify the interpretation of the results. The compressed output data from all channels are collected and compared to determine which IPs or communication channels are affected. The results are summarized in Fig. 5.

Data are checked only in the PS cores. In the case of IP0 and IP1, each of them uses an exclusive output channel, so we cannot know if the error originated in the IP or in the interface. However, IP2 uses a different output channel to each core. In this case, we can distinguish between errors in the IP or errors in the interface. If the error is transmitted to both PS cores, we can assume the error originated in the IP, while an error in the interface should only affect one of the output channels and be observed by one of the cores. Finally, there are errors that affect more than one data set and can be generally assigned to the communication channels.

According to these considerations, we have classified the errors in four categories:

- IP0, ch. 0: errors in IP0 or communication channel 0
- IP1, ch. 1: errors in IP1 or communication channel 1
- IP2, ch. 2 & 3: errors in IP2
- IF (Interface): errors in communication channels

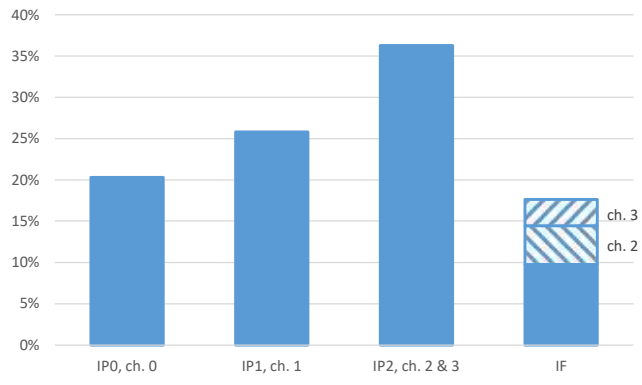

Fig. 5. Location of errors under neutron radiation

In Fig. 5, we can observe that IP2 fails more often than IP1, and IP1 more often than IP0. A possible explanation of this result is that this is caused by the checking order. The results of IP0 are first compared with the results of IP1, and finally the

results of IP2 are compared with the previous ones. Despite the IPs are implemented separately in the programmable logic and the results are sent through parallel communication channels, there are some shared resources in the PS, such as the memory, that may stall processing. As the results of the computations are stored and checked later, they are more susceptible to be corrupted.

The IF errors can be divided in two groups: errors that can be clearly located in one single output channel (channels 2 and 3, filled in Fig. 5 with diagonal stripes pattern), and errors that affect more than one data set. An analysis of the latter reveals that in many cases these errors can be attributed to the input channels, because the wrong output data sets match for the IPs that share the same input channel.

Multiple wrong data sets can also be caused by an SEU or an MCU in the configuration memory of the device that affects more than one IP. However, this is unlikely in comparison with multiple errors associated to channel sharing. As a matter of fact, we did not find any case in which the only modules that do not share any communication channel (IP0 and IP1) failed simultaneously while the other (IP2) was correct. Whenever two or more output data sets were wrong, there was a common input or output communication channel among them.

The results of this experiment show that a TMR architecture is very effective to mask errors, as most data errors affect only one of the redundant computations and can be masked by voting. On the other hand, interface errors amount to 17.6 % of the total errors. Considering only errors in IP2 and the associated output channels 2 and 3, we obtain a similar percentage of interface errors (17.8 %). These figures are significant and may vary according to the relative area of the IP and the interfaces. Thus, the use of redundant architectures to mitigate errors in the input and output data streams is crucial to achieve a good performance of the entire hyperspectral compression system.

VI. Conclusions

In this work, we have evaluated the error sensitivity of the SHyLoC CCSDS-123 IP core for hyperspectral data compression. In contrast to other works, that consider the IP core alone, we focus on system-level evaluation and mitigation, including the data interfaces and the processing subsystem as well. To this purpose we have designed and tested two different architectures that include redundant IP cores, interfaces and processors. These architectures can be implemented using the resources available in commercial MPSoCs. With this approach, we can perform a more practical evaluation, as we can take into account the effects on the compressor's input and output data stream interfaces, and possibly the downstream processor, which will be present in any system using the IP. By using a system-level approach, we were able to validate the robustness of the hyperspectral data compression implementation under neutron and proton radiation. Furthermore, we were able to analyze how errors affect compressed datasets and the relative sensitivity of redundant modules and interfaces.

The experimental results show that the proposed system architectures are effective to mitigate soft errors caused by radiation in the hyperspectral data compression implementation. Cross-section is reduced by about two orders of magnitude in both cases of protons and atmospheric neutrons. Some errors can be corrected by rollback, but most errors are persistent in SRAM-based FPGAs and require a reset or a power cycle to recover. Configuration memory scrubbing may reduce the need to reset and reconfigure the programmable logic.

Acknowledgment

The authors would like to acknowledge ISIS (UK) for providing the neutron beam time and to Centro Nacional de Aceleradores (Spain) for the access to the proton beam. We would like to thank Chris Frost, Carlo Cazzaniga and Maria Kastriotou for their help with the neutron experiment, and Yolanda Morilla, Amor Romero and Pedro Martín-Holgado for their help with the proton experiment.

References

[1] S.-E. Qian, "Introduction to hyperspectral satellites," in Hyperspectral Satellites and System Design. Boca Raton, FL, USA: CRC Press, 2020, ch. 1, pp. 1–52.

[2] A. Zahra, R. Qureshi, M. Sajjad, F. Sadak, M. Nawaz, H. A. Khan, and M. Uzair, "Current advances in imaging spectroscopy and its state-of-the-art applications", Expert Systems with Applications, vol. 238, Part E, Art. no. 122172, March 2024.

[3] R. O. Green et al., "The Earth Surface Mineral Dust Source Investigation: An Earth Science Imaging Spectroscopy Mission," 2020 IEEE Aerospace Conf., Big Sky, MT, USA, Art. no. 2.0101, March 2020.

[4] S. Chabrillat et al., "The EnMAP Satellite - Mission Status and Science Preparatory Activities," 2021 IEEE Int. Geoscience and Remote Sensing Symp. IGARSS, Brussels, Belgium, pp. 123-125, July 2021.

[5] S. Petro, K. Pham and G. Hilton, "Plankton, Aerosol, Cloud, ocean Ecosystem (PACE) Mission Integration and Testing," 2020 IEEE Aerospace Conf., Big Sky, MT, USA, Art. no. 6.0201, March 2020,

[6] M. Rast, J. Nieke, J. Adams, C. Isola and F. Gascon, "Copernicus Hyperspectral Imaging Mission for the Environment (Chime)," 2021 IEEE Int. Geoscience and Remote Sensing Symp. IGARSS, Brussels, Belgium, pp. 108-111, July 2021.

[7] Y. Dua, V. Kumar, R. S. Singh, "Comprehensive review of hyperspectral image compression algorithms," Optical Eng., vol. 59, no. 9, Art. no. 090902, Sept. 2020.

[8] "CCSDS. Lossless Multispectral and Hyperspectral Image Compression, Recommended Standard CCSDS 123.0-B-1; Blue Book" CCSDS:Washington, DC, USA, 2012.

[9] Y. Barrios, A. J. Sánchez, L. Santos and R. Sarmiento, "SHyLoC 2.0: A Versatile Hardware Solution for On-Board Data and Hyperspectral Image Compression on Future Space Missions," IEEE Access, vol. 8, pp. 54269-54287, March 2020.

[10] L.A. Aranda, A. Sánchez, F. Garcia-Herrero, Y. Barrios, R. Sarmiento, J.A. Maestro, "Reliability Analysis of the SHyLoC CCSDS123 IP Core for Lossless Hyperspectral Image Compression Using COTS FPGAs," Electronics, vol 9, no. 10, Art. no. 1681, Oct. 2020.

[11] A. Sánchez, Y. Barrios, L. Santos and R. Sarmiento, "Evaluation of TMR effectiveness for soft error mitigation in SHyLoC compression IP core implemented on Zynq SoC under heavy ion radiation," 2019 IEEE Int. Symp. on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), Noordwijk, Netherlands, pp. 95-98, Oct. 2019.

[12] P. M. Aviles, L. A. García-Astudillo, L. Entrena; M. García-Valderas, P. Martín-Holgado; Y. Morilla; A. Lindoso, "Hardening Architectures for Multiprocessor System-on-Chip," IEEE Transactions on Nuclear Science, vol. 71, no. 8, pp. 1887-1895, Aug. 2024.

[13] M. Berg, "FPGA Mitigation Strategies for Critical Applications," School on the Effects of Radiation on Embedded Systems for Space Applications (SERESSA), Seville, Spain, 2019

[14] Zynq-7000 All Programmable SoC: Technical Reference Manual, Doc. UG585, Xilinx, San Jose, CA, USA, 2016.
[15] P. M. Aviles, A. Lindoso; J. A. Belloch, M. Garcia-Valderas, Y. Morilla; L. Entrena, "Radiation Testing of a Multiprocessor Macrosynchronized Lockstep Architecture With FreeRTOS," IEEE Transactions on Nuclear Science, vol. 69, no. 3, pp. 462-469, March 2022.
[16] W. Grignani, D. A. Santos, F. Viel, L. Dilillo and D. R. Melo, "A Dependable and Low-Cost CCSDS 123 Hyperspectral Image Compressor," IEEE Embedded Systems Letters, doi: 10.1109/LES.2024.3420934 (early access).
[17] A. Tsigkanos, N. Kranitis, G. Theodorou and A. Paschalis, "A 3.3 Gbps CCSDS 123.0-B-1 Multispectral & Hyperspectral Image Compression Hardware Accelerator on a Space-Grade SRAM FPGA," IEEE Transactions on Emerging Topics in Computing, vol. 9, no. 1, pp. 90-103, Jul. 2018.
[18] AMD, "AXI4-Stream FIFO v4.2", LogiCORE IP Product Guide, Vivado Design Suite, PG080, Oct. 30, 2019.
[19] AMD, "Zynq 7000 SoC Technical Reference Manual", UG585 (v1.14), June 30, 2023
[20] R. F. Rice, "Some practical Universal Noiseless Coding Techniques", Jet Propulsion Laboratory, March 1979.
[21] M. D. Berg, K. A. LaBel, and J. Pellish, "Single event effects in FPGA devices 2014-2015," in NASA Electron. Parts and Packaging Program. Electronics Technol. Workshop (NEPP-ETW), 2015, Art. no. GSFC-E-DAA-TN24313. [Online]. Available: https: //ntrs.nasa.gov/api/citations/20150015964/downloads/20150015964.pdf
[22] M. Ceschia et al., "Identification and classification of single-event upsets in the configuration memory of SRAM-based FPGAs," IEEE Transactions on Nuclear Science, vol. 50, no. 6, pp. 2088–2094, Dec. 2003.
[23] F. G. de Lima Kastensmidt, G. Neuberger, R. F. Hentschke, L. Carro and R. Reis, "Designing fault-tolerant techniques for SRAM-based FPGAs," IEEE Design & Test of Computers, vol. 21, no. 6, pp. 552-562, Nov.-Dec. 2004.
[24] K. S. Morgan, D. L. McMurtrey, B. H. Pratt, and M. J. Wirthlin, "A Comparison of TMR With Alternative Fault-Tolerant Design Techniques for FPGAs," IEEE Transactions on Nuclear Science, vol. 54, no. 6, pp. 2065–2072, Dec. 2007.
[25] H. Quinn, A. Watkins, Y. Chen, T. Fairbanks, T. Shepard, and E. Raby, "Recent Results for Commercial Microprocessor Testing," Proc. IEEE Radiat. Effects Data Workshop (REDW), Waikoloa, HI, USA, pp. 47–53, Jul. 2018.
[26] H. Quinn, T. Fairbanks, J. L. Tripp, G. Duran, and B. Lopez, "Single-Event Effects in Low-Cost, Low-Power Microprocessors," Proc. IEEE Radiat. Effects Data Workshop (REDW), Paris, France, pp. 20–28, Jul. 2014.
[27] M. Pignol, "DMT and DT2: Two fault-tolerant architectures developed by CNES for COTS-based spacecraft supercomputers," Proc. 12th IEEE Int. On-Line Testing Symp. (IOLTS'06), Lake Como, Italy, pp. 203–212, Jul. 2006.
[28] H. Quinn, Z. Baker, T. Fairbanks, J. L. Tripp, and G. Duran, "Robust Duplication With Comparison Methods in Microcontrollers," IEEE Transactions on Nuclear Science, vol. 64, no. 1, pp. 338–345, Jan. 2017.
[29] H. Quinn, Z. Baker, T. Fairbanks, J. L. Tripp, and G. Duran, "Software resilience and the effectiveness of software mitigation in microcontrollers," IEEE Transactions on Nuclear Science, vol. 62, no. 6, pp. 2532–2538, Dec. 2015.
[30] E. Chielle et al., "Reliability on ARM processors against soft errors through SIHFT techniques," IEEE Transactions on Nuclear Science, vol. 63, no. 4, pp. 2208–2216, Aug. 2016.
[31] ARM. Cortex-R5 and Cortex-R5F. Technical Reference Manual, Version R1P1 Issue C. [Online]. Available: https://documentationservice.arm.com/static/5f042788cafe527e86f5cc83?token=
[32] K. Greb and D. Pradhan, "HerculesTM microcontrollers: Real-time MCUs for safety-critical products," Texas Instruments, Dallas, TX, USA, White Paper, 2011.
[33] A. B. de Oliveira et al., "Lockstep dual-core ARM A9: Implementation and resilience analysis under heavy ion-induced soft errors," IEEE Transactions on Nuclear Science, vol. 65, no. 8, pp. 1783–1790, Aug. 2018.
[34] M. Peña-Fernández, A. Serrano-Cases, A. Lindoso, S. Cuenca-Asensi, L. Entrena, Y. Morilla, P. Martín-Holgado, and A. Martínez-Álvarez, "Hybrid lockstep technique for soft error mitigation," IEEE Transactions on Nuclear Science, vol. 69, no. 7, pp. 1574–1581, Jul. 2022.
[35] X. Iturbe, B. Venu, E. Ozer, and S. Das, "A triple core lock-step (TCLS) ARM Cortex-R5 processor for safety-critical and ultra-reliable applications," Proc. 46th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. Workshop (DSN-W), pp. 246–249, Jun. 2016.
[36] P. M. Aviles, J. A. Belloch, L. Entrena and A. Lindoso, "Supervised Triple Macrosynchronized Lockstep (STMLS) Architecture for Multicore Processors," IEEE Access, vol. 11, pp. 128706-128723, 2023.
[37] M. Rogenmoser et al., "Design and Experimental Investigation of Trikarenos: A Fault-Tolerant 28nm RISC-V-based SoC", RADiation Effects on Components and Systems (RADECS) Conf. 2024, Maspalomas, Spain, Art. no. F-3, Sept. 2024.
[38] L. A. Tambara, P. Rech, E. Chielle, J. Tonfat and F. L. Kastensmidt, "Analyzing the Impact of Radiation-Induced Failures in Programmable SoCs," IEEE Transactions on Nuclear Science, vol. 63, no. 4, pp. 2217-2224, Aug. 2016.
[39] L. A. Tambara et al., "Reliability-Performance Analysis of Hardware and Software Co-Designs in SRAM-Based APSoCs," IEEE Transactions on Nuclear Science, vol. 65, no. 8, pp. 1935–1942, Aug. 2018.