Impact of Neutron-Induced SEU in FPGA CRAM on Image-Based Lane Tracking for Autonomous Driving: From Bit Upset to SEFI and Erroneous Behavior

Tomonari Tanaka, Wang Liao, *Member, IEEE*, Masanori Hashimoto, *Senior Member, IEEE*, and Yukio Mitsuyama, *Member, IEEE*

***Abstract*— Configuration random access memory (CRAM), which consists of the static random access memory susceptible to single event upset (SEU), configures all of the user logic in field-programmable gate array (FPGA). In this article, we evaluate the impact of SEU in CRAM on the image-based lane tracking for autonomous driving via neutron irradiation experiments. In the experiments, the cross section of bit upsets and its corresponding single event function interrupt (SEFI) in the logic function of image processing are measured. By using a virtual driving environment, we observe whether a bit upset finally induces a severe SEFI of the system failure. The system failure is observed by an abnormal behavior on the virtual autonomous car driving by lane tracking. Experimental results show the bit upset has a maximum probability of 23% to induce SEFI and finally 8% of the bit upsets lead to failures in lane tracking. All the SEFIs observed in irradiation experiments are reproduced in fault injections with the same bit address. The cross section of SEFI was estimated in fault injections with reasonable precision. Moreover, we evaluate the improvement of system reliability by the error correction against soft errors. Our evaluation result shows that the error correction within the period of two-frame processing time could reduce the 74% SEFI cross section, but it has little benefit for system failure cross section if an error happens during the period of the image processing.**

***Index Terms*— Configuration random access memory (CRAM), erroneous behaviors, field-programmable gate array (FPGA), single event function interrupt (SEFI).**

I. Introduction

SOFT errors induced by neutrons, which threaten the reliability of terrestrial semiconductor devices, is attracting attention in the automobile industry. In 2011, ISO26262 [1] first included the test of the random failure, which mainly originates from soft errors, as a part of the quality examination of hardware and software used in automobiles. Besides the traditional sensors and controllers, in the coming era of autonomous driving, it is no doubt that the reliability of devices for image processing will also become a major concern. Due to its ability of parallel processing, field-programmable gate array (FPGA) is one of the most promising devices to achieve high-performance image processing. Therefore, the evaluation of soft errors in FPGA is important for autonomous driving.

At present, the mainstream of commercial FPGAs utilizes static random access memory (SRAM) as configuration random access memory (CRAM) to implement user logic. Once a bit upset in a critical part of CRAM, the user logic will be interrupted, i.e., an occurrence of a single event function interrupt (SEFI), and not restored until the next reconfiguration. Meanwhile, similar to other digital circuits, FPGA also contains block RAMs (BRAMs), flip flops (FFs), and combination logic gates. Therefore, for FPGAs, the issue of single-event upsets (SEUs) is a major concern. With the evaluation on bit upset in FPGAs using irradiation experiments, the vendors of FPGAs, e.g., Xilinx [2], usually provide the cross section of the bit upsets in RAM cells. The error rate in the memory elements was also evaluated using the readback method or simple mathematical computations [3], [4]. SEFI of application on microprocessors was evaluated in [4], focusing on the software codes of CoreMark and advanced encryption standard (AES). The other characterizations of the bit upsets, such as the energetic dependence [5], [6], were also measured by researchers for their specific aims.

As the evaluation on a higher level of error, i.e., SEFI, lots of work has been conducted for some general applications such as processor and matrix multiplication including, but not limited to [7]–[11]. On the other hand, as more highly application-specific designs for the autonomous driving system implemented in FPGA, some FPGA-based applications of image processing, which is highly related to autonomous driving applications, were evaluated in the previous work [12]–[18]. These evaluated applications could be categorized into two types. The first type includes a convolutional neural network (CNN) [12]–[16]. The reliability of the final classification result was mainly focused in the evaluation with irradiation experiments [13]–[16] and the fault injection [12]. In these works, the probability of SEFI occurrence in each

Manuscript received October 19, 2021; revised November 22, 2021; accepted November 24, 2021. Date of publication November 30, 2021; date of current version January 19, 2022. This work was supported in part by the Japan Science and Technology agency, Program on Open Innovation Platform with Enterprises, Research Institute and Academia (JST-OPERA) Program, Japan, under Grant Number JPMJOP1721; and in part by the JSPS Grants-in-Aid for Scientific Research (KAKENHI), Japan, under Grant Number JP16K00083 and Grant JP21K17721.

Tomonari Tanaka is with the Graduate School of Engineering, Kochi University of Technology, Kami, Kochi 782-8502, Japan.

Wang Liao and Yukio Mitsuyama are with the School of Systems Engineering, Kochi University of Technology, Kami, Kochi 782-8502, Japan (e-mail: liao.wang@kochi-tech.ac.jp; mistuyama.yukio@kochi-tech.ac.jp).

Masanori Hashimoto is with the Department of Communications and Computer Engineering, Kyoto University, Kyoto 606-8501, Japan (e-mail: hashimoto@i.kyoto-u.ac.jp).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNS.2021.3131346.

Digital Object Identifier 10.1109/TNS.2021.3131346

layer [14], the difference of reliability between standard, quantized, and triple modular redundancy (TMR) CNNs [16], and the influence of data type used in calculation [15] is also discussed. Meanwhile, image filters, such as the Median filter and the Gaussian filter, are the second type of evaluation target [17], [18], mainly evaluated using fault injection.

As stated above, the previous work mainly evaluated bit upset-induced SEFI within the layer of image processing in autonomous driving. However, the SEFI in its upper layer of the whole autonomous driving system, i.e., SEFI of the malfunction of autonomous driving, is also worth to be investigated. In this work, we evaluated an autonomous driving application based on lane tracking, which includes the least necessary modules of the image filters used in canny edge detection and the calculation of lane position. Combining with the virtual driving environment of Gazebo, we observed the severity of bit upset-induced SEFIs in both irradiation experiment and fault injection. We further classify the SEFIs according to their severity into the minor SEFI limited to the layer of image processing, i.e., the function of lane tracking still works well with the errors, and the severe SEFI in the upper layer of autonomous driving, i.e., the malfunction of lane tracking. Here, we define such severe SEFI as system failure throughout this article. The system failure is observed by the abnormal behavior on the virtual autonomous car driving by lane tracking. This work aims to advance the knowledge about how probable and by what mechanism the bit upset in CRAM would influence the reliability of autonomous driving applications based on image processing implemented in FPGA.

This article is organized as below. The design of the image-based lane tracking system and the error monitoring method are described in Section II. Next, Section III explains the setup for irradiation experiments including the implementation of the lane tracking in a virtual driving environment during the experiment. In the same section, the experimental result of cross section of bit upsets, SEFIs, and system failures are also discussed. Section IV evaluates the cross section of SEFI and system failure and the improvement on the reliability of the system with the error correction using fault injection. Finally, Section V concludes this article.

II. Design Under Test and Its Error Monitoring

In this article, we evaluate the impact of SEU in FPGA CRAM on image-based lane tracking for autonomous driving. To focus on the image processing, the design under test (DUT) implemented in FPGA includes only lane tracking that processes the input image to obtain the center coordinate of the edge lane. Host PC and the peripheral circuits/modules deal with the rest tasks of steering the car driving according to the center coordinate of the edge lane, generating the virtual image for lane tracing, and the other functions related to data transferring. In this section, we introduce the DUT and the modules for monitoring errors in DUT. Both the DUT and monitoring modules are implemented by programmable logic (PL) in FPGA.

A. DUT of Lane Tracking Based on Canny Edge Detection

To achieve the function of lane tracking for autonomous driving, the DUT mainly calculates the coordinate of the lane edge in the input image. The general information on this design is given here, while the details are described in our previous work [19].

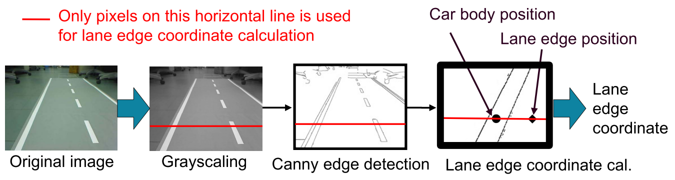
Only pixels on this horizontal line is used for lane edge coordinate calculation
Car body position
Lane edge position
Lane edge coordinate
Original image
Grayscaling
Canny edge detection
Lane edge coordinate cal.

Fig. 1. Image processing flow in our DUT. The lane edge coordinate/position is calculated based on canny edge detection. Although the full image is processed, only the pixels on the horizontal line are utilized to calculate lane edge coordinate.

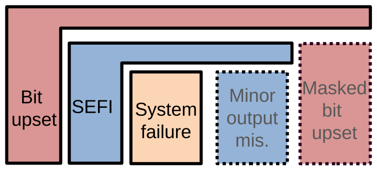

Fig. 2. Three levels of errors: bit upset, SEFI, and system failure. The operation of lane tracking system could be observed in the virtual environment. Therefore we added an SEFI of upper level, namely system failure.

Fig. 1 shows the processing flow of the image. For edge detection, the widely-used algorithm of canny edge detection is used [20]. As shown in Fig. 1, at the red horizontal line, we searched the pixels being judged as the edge lane and calculated their center coordinate. After calculating the coordinate of the lane center in image processing, the action decision will be made to keep the distance between the lane edge position and the current car body position within a certain range. An action of steering left or right will be taken to track the lane edge properly if the distance is out of the given range.

The DUT is divided into three modules in logic design and is synthesized by high level synthesis (HLS) of Xilinx Vivado development suite without any radiation hardening option since our target is a terrestrial application. To collect more error events, three DUTs are implemented with 3× CRAM bits in the FPGA. The DUTs are implemented in the PL of Xilinx XC7Z020-1CLG400C of Zynq 7000 series all programmable system on chip (APSoC) in 28-nm technology node.

B. Monitoring Errors in DUT

During irradiation experiment and fault injection in the DUT of the lane tracking system, three levels of errors are defined: bit upset, SEFI and system failure as shown in Fig. 2. The bit upset includes single bit upset (SBU), multiple cells upset (MCU), and multiple bit upset (MBU) in CRAM. Here, MCU refers to SBUs in several words, while MBU refers to several bits upset inside the same word due to a single event. On the other hand, we define SEFIs as the damage in logical function, e.g., repeated mismatch of outputs from modules compared to the golden one due to the bit upsets. Finally, if such repeated mismatches cause the abnormal operation of being unable to track the lane, a severe SEFI of system failure is determined. We could observe the operation of the lane tracking system in the virtual environment, which will be introduced in Section III. Both system failures and other SEFIs (minor output mismatching) could be also observed from the mismatched output from the golden one. The difference

TABLE I
RESOURCE UTILIZATION OF THE DUT1. THE DUT2 AND DUT3 USE ONLY ONE AND FOUR MORE LUT(S), RESPECTIVELY, COMPARED TO THE DUT1

Resource	LUTs	FFs	Block RAMs	DSPs
Available	53200	106400	140	220
Gray.	347	466	0.5	3
Canny.	2869	3847	5	5
Cal.	1495	1626	0.5	3
Total	8.72 %	5.28 %	4.29 %	5.00 %

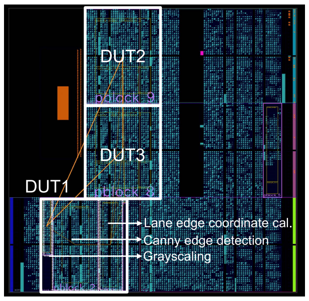

Fig. 3. Floorplan in Zynq 7000 Z7020 used in both irradiation experiment and fault injection. The DUTs are placed inside the square blocks to calculate address of bits of CRAM used in DUTs.

is whether the mismatches are severe enough to cause the abnormal operation of being unable to track the lane or not.

To monitor the bit upset in CRAM, soft error mitigation (SEM) intellectual property (IP) [21] is utilized and implemented independently of the DUT. During the DUT processing of the image, SEM IP will work continuously to check each 32-bit word in CRAM with cyclic redundancy check (CRC) and is capable of single-error correction and double error detection (SECDED). In our design, the SEM IP is provided with a 100 MHz clock so that it can scan the whole CRAM of 25.7 Mbit within 8.0 ms. If bit upset is detected, the SEM IP will report the address of the error bit. On the other hand, the range of bit address of CRAM utilized in the DUT could be calculated by the tool of automatic configuration memory error-injection (ACME) [10]. Therefore, we can find the relation between bit upset and SEFI by analyzing whether the address of the upset falls within the range of the bit addresses of the module suffering the SEFI or not.

To calculate the range of bit addresses used in DUT, the DUTs are placed within a square block in the floorplan. Fig. 3 shows our placement including the DUTs in the floorplan used in both irradiation experiment and fault injection. Although the placement in each block is different, the logic design of all the DUTs is identical. To test all the DUTs fairly in irradiation experiments, we adjust the block size to keep the percentage of utilization of CRAM 70% in these three DUTs. The resource utilization of DUT1 is shown in Table I, while the DUT2 and DUT3 only used one and four more look up table(s) (LUT(s)), respectively, compared to the DUT1. In the area beyond the block of DUTs, other user logic for SEFI monitoring modules and receiving images from the host PC described below are placed by the synthesis tool automatically. It should be noticed that placing the whole DUT at such a limited block in a high density could increase the occurrence probability of MCU within the DUT. We will touch on this issue in Section IV-B.

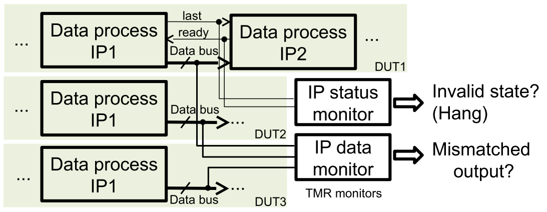

Fig. 4. Monitor modules for DUTs. We monitor status signals and data buses, respectively, to distinguish an unintended shutdown with calculation error in the module during SEFI. All modules are triplicated to prevent itself suffering from soft errors.

For observing SEFI, we design the monitoring modules to monitor the input and output signals between the modules inside the DUTs. The logic designs of each DUT consists of *grayscaling (gray.)*, *canny edge detection (canny.)*, and *lane edge coordinate calculating (cal.)*. The image data are transmitted from the upstream module to the downstream module with the protocol of Advanced eXtensible Interface (AXI) Stream, which mainly contains status signal and data bus. By monitoring the status signal and data bus, respectively, as shown in Fig. 4, we could distinguish between the stopping working and incorrect calculation of the modules during SEFI. We monitor the status signals of "ready" and "last" in the AXI Stream interface standing for ready to receive and finishing transmission of image data. If either ready or finish signal is not observed at all during the processing/transmission of a single image data, we can confirm that the related module in DUT does not finish the transmission of the image normally and stays in a hang. On the other hand, we monitor the data bus of each module with comparison to their counterpart in the other DUTs. If the comparison result shows different data in one DUT repeatedly, we can confirm that the module in DUT is suffering SEFI of outputting data incorrectly. The feedback of monitoring modules is encoded such that the location of SEFI could be identified at the module level. All the monitoring modules are designed with triple module redundancy (TMR) to prevent the failure inside the monitoring module itself.

As for observing system failures, the virtual autonomous driving based on the image processing in DUT is performed as will be described in Section III. We discriminate a system failure if the car failed to run along the lane edge line.

III. IRRADIATION EXPERIMENT

A. Experiment Set-Up

We utilize an open-source robot simulator Gazebo [22] running on a host PC with Linux OS to build the virtual driving environment including the function of providing a real-time camera image, receiving the decision of action, and moving the car according to the decision. Fig. 5 shows an overview of the virtual driving environment built in Gazebo. The map for car driving is referred to the FPGA design competition of autonomous driving held at the international conference on field-programmable technology (ICFPT) 2019. We design a simple two-wheel robot car with a virtual camera on its left

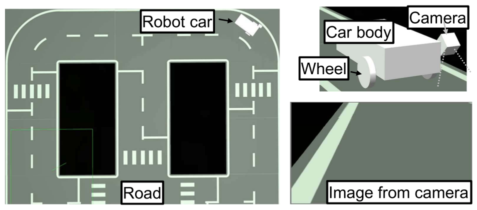

Fig. 5. Virtual driving environment running on host PC with linux OS. The virtual car has a simple two-wheel car with a virtual camera on its left side for supervising lane edge.

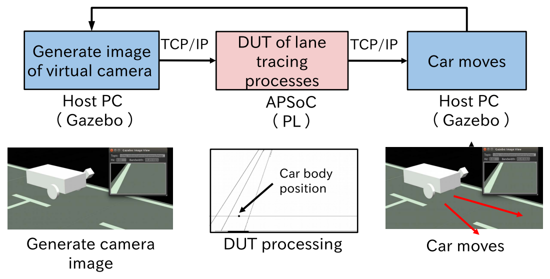

Fig. 6. Test flow of each run.

side for supervising lane edge. The image from the virtual camera has a resolution of 640 × 480.

The test flow is shown as Fig. 6. At the beginning of an irradiation test, all the CRAM bits are rewritten to ensure that DUTs and peripheral logic are implemented correctly. Simultaneously, the virtual driving environment will also be reset including moving the car to the start point and forcing it to wait until the first steering action decision based on the coordinate of the center of the lane edge calculated in DUT. Our decision policy for steering action is simple: we try to keep the coordinate of the center of the lane edge in an expected range. When we obtained the current coordinate calculated by image processing in DUT, we compare it with the current car position and decide which action among going straight, steering left and right to take. After steering the car for around 1 ms, Gazebo will consequently stop the movement of the car, generate the new camera image, and send it to APSoC via transmission control protocol (TCP) package. When an image data arrives at the processor (PS) of APSoC, it will be stored to dynamic random access memory (DRAM) and then DUTs in PL could fetch the data by direct memory access (DMA). After DUTs finish processing, PS sends back the coordinate of the center of the lane edge, the action decision, and the feedback of monitoring modules of each DUT. At the end of the single loop in a test run, Gazebo on the host PC receives the readback data from PS and moves the car accordingly.

During an irradiation test, the DUT works at 100 MHz and has the performance of processing 60 frames per second. However, due to the extra time cost of transferring images from the host PC to DUT, three frames of images are sent to DUTs per second for lane tracking. If no upset in CRAM for a long time, the test finishes at the 700th frame. However, if bit upsets are detected in CRAM, the test run will end in 200 frames for detecting SEFI occurrence. Besides, we check whether a bit flip has occurred in CRAM when a SEFI is detected, but the number of upsets is not counted because the uncorrected error would make the SEM IP stop working. That means the accumulation of bit flip might happen during the observation period of 200 frames. The possibility of error accumulation will be discussed in Section III-B.

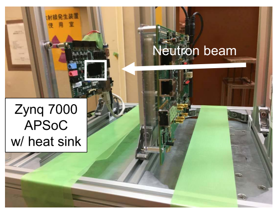

Fig. 7. Set-up of board under test. Both PS and PL of APSoC are irradiated.

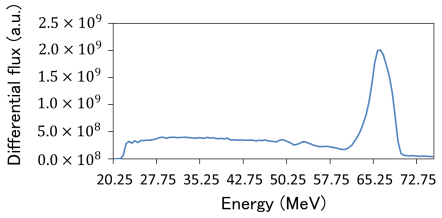

Fig. 8. Beam spectrum of quasi-monoenergetic neutron beam produced by 70 MeV proton in CYRIC at Tohoku University.

The host PC for Gazebo is placed inside the monitor room shielded from the neutron beam, and therefore it is supposed that no errors would occur in PC. Meanwhile, the board of APSoC is placed along the beamline as shown in Fig. 7, and the whole PS and PL are irradiated by neutrons. Indeed, some undesirable errors, e.g., data corruptions in DRAM or BRAM, are undetectable. However, the image data on a DRAM is rewritten for each frame and only temporal data is stored in BRAM in our application, such errors are less probable to occur repeatedly while we observe the DUTs using the monitoring modules. Therefore, we could distinguish the undesirable errors from the SEFIs in DUT.

As for the neutron source, a quasi-monoenergetic neutron test was conducted at Cyclotron and Radioisotope Center (CYRIC) at Tohoku University [23]. The neutron beam is produced by a 70 MeV proton source, and it has a flux peak at the energy near 70 MeV as shown in Fig. 8 of the beam spectrum. The flux at the place of APSoC board is 5.4×10^4 n $\cdot$ cm$^{-2} \cdot$ s^{-1}.

B. Experimental Results

Table II shows the experiment results using the neutron source. We ran the test for measuring cross sections of bit upsets and SEFI in two separate rounds. In the test round of 4.1 h, bit upsets were measured individually using SEM IP with a single error correction. In the other round, the function of error correction was disabled to observe SEFIs in the unhardened system.

For the bit upsets, as we could observe in Table II, the cross section of total bit upset events is 5.92×10^{-15} cm^2/bit, which is within the error margin of the value reported by Xilinx [2]. According to the measured cross section of bit upset, 0.07 errors can be accumulated on average during the period of 200 frames. From this estimation, we think that the accumulation concern described in Section III-A is negligible.

TABLE II
MEASUREMENT RESULT FROM IRRADIATION EXPERIMENT USING THE NEUTRON SOURCE

Error type		Test period (h)	Fluence ($n \cdot cm^{-2}$)	# of bits in test	# of events	Cross section ($cm^2 \cdot bit^{-1}$)	Error margin
Bit upset (individual test)	total	4.1	7.96×10^8	25.7 M	121	5.92×10^{-15}	± 9.09 %
	SBU	4.1	7.96×10^8	25.7 M	101	4.94×10^{-15}	± 9.95 %
	MCU	4.1	7.96×10^8	25.7 M	14	6.84×10^{-16}	± 26.7 %
	MBU	4.1	7.96×10^8	25.7 M	6	2.93×10^{-16}	± 40.8 %
SEFI		13.4	2.44×10^9	6.35 M	21	1.35×10^{-15}	± 21.8 %
System failure		13.4	2.44×10^9	6.35 M	7	4.52×10^{-16}	± 37.8 %

On the other hand, the comparison of the cross sections between SEFI and bit upset shows that the cross section of SEFI is of the same magnitude as that of bit upset. The probability of bit upset inducing a SEFI would be as high as 23%. Moreover, we also observed that around 33% of SEFIs lead to the system failure of lane tracking in the virtual driving environment. To identify the cause of system failure, we examined the location and time of the occurrence of the observed SEFIs and bit upsets. We confirmed that the time and location of the bit upset and the SEFI are identical for more than half of the observed events. Thus, the bit upset of CRAM in these events induced the SEFI with high probability.

IV. FAULT INJECTION

Due to bit upsets probably being masked, observing SEFI and system failure requires a large number of bit upsets, but the beam time is limited. As a complement to the irradiation test, fault injection by intentionally flip bits in the CRAM is an efficient way to allow us to evaluate the vulnerability of bit upset-induced SEFIs. In this section, we conduct fault injection and calculate the device vulnerability factor (DVF) of the image-based lane tracking system, which represents the probability of an upset bit inducing the SEFI [24]. Combining with DVF obtained in the fault injection and measured bit upset cross section in irradiation test, we further estimate the cross section of SEFI and system failure and compare them with the measurement result to validate the estimated value. Furthermore, we investigate the benefit of the reliability improvement of DUTs brought by error correction techniques. The same implementation of DUTs are used for both fault injection and irradiation experiments.

Before injecting a large number of errors, we reproduced the runs, where SEFI was observed in the experiment, by injecting fault to the same address. The reproduction validates the same results in both experiments and fault injection. In the following parts of this section, we will focus on the details and discussion on fault injection for calculating DVF and estimating SEFI cross sections.

A. Flow of Evaluation on DVF

We conducted the injection on three modules in DUT1 to calculate the DVF. The targets of error injection are the CRAM addresses of the bits for the logical functions in DUT1, which are hereafter called essential bits. In contrast, the bits are nonessential bits, which were not utilized in logical function and are supposed not to affect the system even if the bit-upset occurs. Furthermore, if the upset in an essential bit induced a SEFI, such an essential bit is a critical bit for the designed system. The DVF is calculated with the following equation:

$$\mathrm{DVF} = \frac{N_{\mathrm{CB}}}{N_{\mathrm{EB}}} \tag{1}$$

where N_{EB} and N_{CB} are the number of essential bits and critical bits in the DUT1, respectively. Similarly, this definition is also applicable to system failure, where the critical bit induces system failure. We will discuss these two kinds of DVF below, but it should be noted that the DVF of system failures is a subset of the DVF of SEFIs, as shown in Fig. 2.

TABLE III
REQUIRED NUMBER OF THE INJECTIONS N_{INJECT} TO EACH MODULE IN DUT1. N_{EB} IS THE NUMBER OF THE ESSENTIAL BITS

module	Gray.	Canny.	Cal.
N_{EB}	84,643	889,951	419,148
N_{inject}	8,625	9,501	9,389

According to the ACME tool, the DUT1 has 20 12 199 configuration bits (which includes 13 96 106 essential bits). Due to a large number of configuration bits, exhaustive testing on all the essential bits is time-consuming. Instead of exhaustive testing, we adopted the statistical fault injection by flipping the sampled bits in CRAM randomly. To satisfy the confidence level and error margin of the statistical test, the required number of the sample bits for injecting error is calculated by the following equation proposed in [25]:

$$N_{\mathrm{inject}} = \frac{N_{\mathrm{EB}}}{1 + e^2 \cdot \frac{N_{\mathrm{EB}}-1}{t^2 \cdot p \cdot (1-p)}} \tag{2}$$

where N_{EB} is the total essential bits of the design, e is the margin of error, t is the confidence level, and N_{inject} is the required number of injections on N_{EB}. To balance the error number and statistical significance, a confidence level of 95% and error margin 1% were selected. N_{EB} and N_{inject} for each module are shown in Table III. As mentioned in Section II-B, "gray.", "canny.", and "cal." stand for the modules of canny edge detection, grayscaling, and lane edge coordinate calculating, respectively.

In the fault injection, we assumed the occurrence of SBUs, which means that we injected error to only one of the selected addresses during each run of virtual driving. The set-up of a virtual driving environment for fault injection is completely the same as the irradiation experiment described in Section III. In contrast to random timing of error occurrence in the irradiation experiment, the error was injected by SEM IP at the fixed tenth frame after the start. After error injection, we observe 200 frames as we did in the irradiation experiment. When one run is completed, all the bits of CRAM are completely

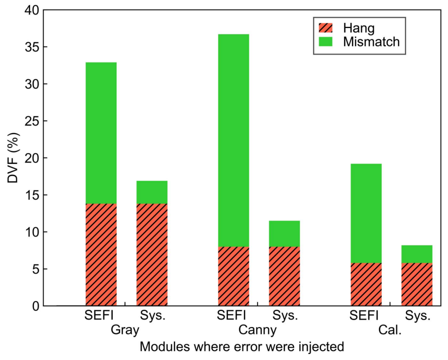

Fig. 9. DVF on the fault injection in essential bits, where "hang" is the invalid state of the module, and "mismatch" is the incorrect data transferring.

rewritten and repeat the run until all the selected addresses are evaluated.

B. Result of DVF and Estimation on Cross section of SEFIs and System Failures

Fig. 9 shows the DVF of SEFIs and system failures obtained by fault injection to the essential bits in each module. The sum in the y-axis of stacked bars is the total DVF of each module. On the other hand, because we could observe a relatively large amount of the total errors in fault injection, we classified the bit upset-induced SEFIs into two types: the module with injected error falling into a hang and the mismatching output with golden circuit i.e., incorrect calculation of the module. We discriminate the type of SEFIs by the monitoring results of status and data signals from the AXI Stream interface of the module as stated in Section II-B. As the result, in Fig. 9, the stacked bar of red and green, labeled as "hang" and "mismatch," stand for the DVF of the hang and the mismatched output, respectively. From this figure, we could observe that all the SEFI events of hangs finally induced the system failures, while the SEFIs of mismatch led less probably to the system failures. This could be attributed to the module in hang state would stop any calculation and data transmission and thus, the whole system would lose its function and fail to track the edge lane. As the SEFIs of hangs have a proportion of 22% to 41%, such type of SEFIs has a large impact on the system reliability.

As the exception of the above results, in a very small probability, we confirmed that the ten bits cause the stuck of the PS. When these bits were flipped, the OS on the PS completely froze. We suppose the reason is that such bits caused an illegal operation of DRAM by the AXI Video DMA (VDMA) IP, which is used for fetching the image data from PS to our DUT in PL [26]. A similar result was reported by Fleming and Thomas [27]. Since it has a main impact on PS and its proportion is small, we did not include such part in our calculation. On the other hand, we also confirmed that some injected errors induced the SEFIs in the untargeted module such as modules of DUT2 and DUT3 and monitoring modules. These SEFIs were induced by up to 0.6% of fault injection. We suppose that these bits are not in the successive addresses and could not be considered by the ACME tool correctly. But since the number of such bits is also small, we still exclude them from the DVF calculation.

TABLE IV

REQUIRED NUMBER $N_{\text{INJECT-NONEB}}$ OF THE INJECTIONS TO THE NONESSENTIAL BITS. N_{NONEB} IS THE NUMBER OF THE NONESSENTIAL BITS IN EACH MODULE

module	Gray	Canny	Cal.
N_{nonEB}	23,584	396,417	194,964
$N_{inject-nonEB}$	6,825	9,377	9,153

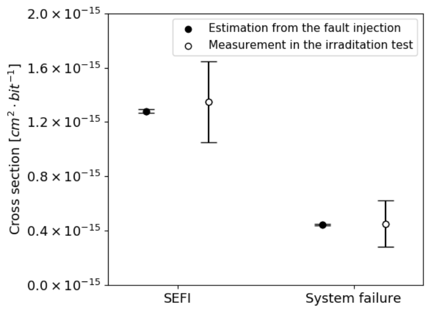

Fig. 10. Measured and estimated cross sections of SEFIs and system failures.

Next, for estimating cross section of SEFIs and system failures from fault injection, the DVF_{DUT} is calculated by

$$\text{DVF}_{\text{DUT}} = \frac{\sum \text{DVF}_i \cdot N_{\text{EB}_i}}{\sum N_{\text{EB}_i}} \tag{3}$$

where i stands for the index of each module. The DVF_{DUT} is essentially the proportion of the critical bits to the total essential bits in DUT. For SEFI and system failure, DVF_{DUT} is 30.6% and 10.1%, respectively, which means 33.0% of SEFIs lead to system failures.

Utilizing the DVF_{DUT} obtained from fault injection, we estimate the cross section of SEFIs and system failures using

$$\begin{aligned}\sigma(\text{type}) &= \sigma(\text{bit upset}) \times \text{DVF}_{\text{type}} \times \frac{N_{\text{EB}}}{N_{\text{EB}} + N_{\text{nonEB}}}\\ &= \sigma(\text{bit upset}) \times \frac{N_{\text{CB}_{\text{type}}}}{N_{\text{EB}} + N_{\text{nonEB}}}\end{aligned} \tag{4}$$

where type of cross section and DVF is either SEFIs or system failures, and $\sigma(\text{bit upset})$ is the cross section of bit upset measured in the irradiation experiment. In this equation, we assume nonessential bits will completely not induce any SEFIs. To confirm this assumption, we also conducted the fault injection to nonessential bits of the numbers shown in Table IV, which also ensures a confidence level of 95% and an error margin of 1%. We confirmed no SEFIs by injecting errors to the nonessential bits.

Fig. 10 shows the cross section for SEFIs and system failures in the fault injection and the irradiation experiment. The error bars stand for one standard deviation. This figure shows that the estimated value is within the error margin of the measured value in the irradiation experiment. Therefore, from the comparison between the irradiation experiment and the fault injection, the result shows the impact of soft errors on the systems could be evaluated within a small error in the proposed virtual driving environment.

Furthermore, we considered the influence of MBU/MCU events on the DUT placed in the given block described in Section II-B. The fault injection targeted only SBU, although the irradiation experiment includes the effect of MBU/MCU. We confirmed that these experimental results have matched within the error margin as shown in Fig. 10. However, we think that the effect of MBU/MCU events depending on the floorplan of FPGA implementation is worth investigating in the future.

C. Improvement of System Reliability by Error Correction

Currently, vendors of SRAM based FPGA tend to provide soft error correction tools [28], including the SEM IP used in this work. In this section, we evaluate the improvement of system reliability by such on-chip correction tools.

For the evaluation of the system reliability for the image-based lane tracking with error correction in an FPGA, the selected essential bits are the same as the previous experiment of fault injection to the system without error correction. We repeated the injection flow described in Section IV-A except a newly added step of the error correction. As for the error correction, we will flip the error bit back to its original value in a fixed period of delay time after injecting the error, instead of enabling the built-in correction of the SEM IP. The utilization of this alternative correction method is because enabling the correction function of SEM IP requires a resynthesis with changing the addresses of essential bits.

Before showing the DVFs of the system with error correction, let us explain the delay before correcting the error bit. As stated above, because we flip the error bit manually, the delay time, which is the time for error detection and correction by the on-chip correction module, should be determined before evaluation on DVF of the system.

In our tested devices, the error correction tool, namely SEM IP, works at 100 MHz and will detect and correct the single bit error in a delay of 9 ms at most [21]. More specifically, it takes up to 8.0 and 0.6 ms for detection and correction, respectively. Here we considered the worst case of the latency. It should also be noticed that SEM IP and the DUTs work parallelly, which means the DUT would still process the data during the SEM IP detect and correct the errors. Fig. 11 shows the image processing flow in the time axis, under the supposed environment of the practical driving system with the assumption of the camera at the frame rate of 60 fps. As shown at the top two axis lines in this figure, since the image processing of lane tracking in DUT takes 6.3 ms, which is less than the time of 16.7 ms (60 fps) for transferring an image from a camera, the image processing would become idle state while waiting for the image of the next frame. If an SBU occurred, two cases exist as shown at the two bottom lines in Fig. 11. In the first case, the SBU happens shortly (within 1.4 ms) after the DUT entering an idle state. Therefore, the error could be corrected before the image processing. Meanwhile, in the second case, the SBU happens at other timing besides the timing of the first case and one frame of the image was being processed with error uncorrected.

To reflect the first case that the processing of an image is not influenced by a bit upset in the error injection, we would stop the processing of DUT, flip the target essential bit, then flip it back in 9 ms, and finally restart the DUT. It should be noted that during the period of error being uncorrected, although the DUT stops processing the image, the influence may still be brought into the DUT due to the upset of some bit, e.g., the CRAM bits related to external routing. In contrast, since one frame of image is influenced by a bit upset, we would flip the target bit at a random timing during the processing, and flip it back immediately before the start of the transferring of the next frame.

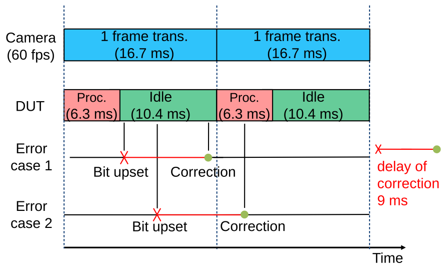

Fig. 11. Supposition of the error occurrence in the actual machine using 60 fps camera and SEM IP, where "case 1" is the situation that the bit upset and error correction occurred in the idle state of the DUT, i.e., not doing calculation while the time from the bit upset to the correction, and "case 2" is the situation where the DUT is affected by the bit upset during the calculation.

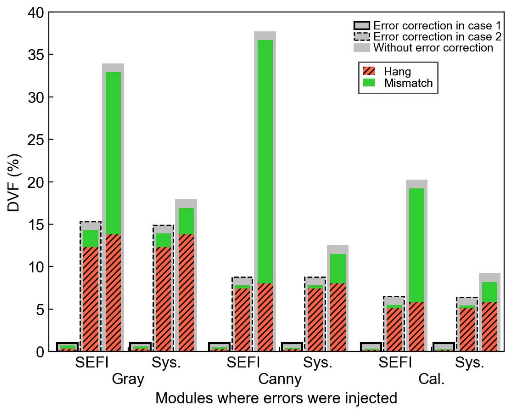

Fig. 12. DVF under the fault injection with error correction, where "hang" is the invalid state of the module, and "mismatch" is the incorrect data transferring.

When the SBU occurs in the system, it could be either case of none (the first case) or one (the second case) frame being influenced. This means the real cross section or DVF of bit upset-induced SEFI should be between those of cases one and two. To show the improvement impartially in these two cases, we conducted the error injection with the same and full set of the sampled essential bit for the first and second cases separately to calculate their DVFs independently.

Fig. 12 shows the DVF of the system with the error correction in the case of none (the first case) and one frame (the second case) being influenced, as well as the DVF of the system without the error correction as the reference. From this figure, we could observe a significant benefit in the DVF reduction of SEFIs from the error correction in either case compared with the reference of errors uncorrected. However, when it comes to a system failure, the trend of reduction of DVF becomes different in the two cases. The reduction is not obvious in the DVF of system failures if the error is not corrected before DUT starting processing in the second case. Meanwhile, if the error is corrected before processing, the DVF reduction is still significant as shown in the result of the first case. Obviously, in the figure, the key difference of these two trends could be attributed to the SEFIs of the type "hang." The results of DVF suggest two facts, when an upset occurs in the critical bit as a source of "hang:" 1) the module, which is working, would fall into a hang immediately and stop working and 2) such hangs are irreversible since no effects after correcting it. As a hint for the system design of high reliability, it might be important to ensure that the correction is fast enough to correct an error during the idle state of circuits.

In addition, the ten reported bits in Section IV-B, which lead to the malfunction of the OS on the PS, were confirmed with no functional error in the first case. This is consistent with our assumption that such bits would lead to an illegal operation of DRAM by VDMA IP since VDMA IP stops working when an error happens in the first case.

V. Conclusion

This article discussed the impact of SEU in CRAM on image-based lane tracking implemented in FPGA. We focused on revealing the relationship between the bit upset, SEFI, and severe SEFI of the malfunction of the whole autonomous driving application, namely system failure, in a virtual driving environment. The irradiation test and the fault injection were performed on the same DUTs and their implementation. The result from the irradiation test shows around 23% of the bit upsets would induce the SEFIs, and 33% of the SEFIs would finally induce system failures. Moreover, we estimated the cross section of SEFIs using the fault injection. The cross section of SEFIs and system failures from the irradiation experiment and the fault injection were compared. The comparison result shows a good precision of the evaluated cross section. In addition, we investigated the improvement in the reliability of the lane-tracking system by the error correction. Our investigation shows a large reduction of SEFIs probability by the error correction. In order to reduce the probability of system failures drastically, the error should be corrected during the idle state of DUT.

References

[1] *Road Vehicles—Functional Safety—Part 1 to Part 10*, Standard ISO 26262:2011, Geneva, Switzerland, 2011.

[2] *Device Reliability Report*, Xilinx, San Jose, CA, USA, UG116, 2nd Half 2019, Apr. 2020.

[3] V. Vlagkoulis *et al.*, "Single event effects characterization of the programmable logic of Xilinx Zynq-7000 FPGA using very/ultra high-energy heavy ions," *IEEE Trans. Nucl. Sci.*, vol. 68, no. 1, pp. 36–45, Jan. 2021.

[4] H. Quinn, T. Fairbanks, J. L. Tripp, G. Duran, and B. Lopez, "Single-event effects in low-cost, low-power microprocessors," in *Proc. IEEE Radiat. Effects Data Workshop (REDW)*, Jul. 2014, pp. 20–28.

[5] D. S. Lee *et al.*, "Single-event characterization of 16 nm FinFET Xilinx UltraScale+ devices with heavy ion and neutron irradiation," in *Proc. IEEE Nucl. Space Radiat. Effects Conf. (NSREC)*, Jul. 2018, pp. 275–282.

[6] C. Cai *et al.*, "SEE sensitivity evaluation for commercial 16 nm SRAM-FPGA," *Electronics*, vol. 8, no. 12, p. 1531, Dec. 2019.

[7] L. A. Tambara, P. Rech, E. Chielle, J. Tonfat, and F. L. Kastensmidt, "Analyzing the impact of radiation-induced failures in programmable SoCs," *IEEE Trans. Nucl. Sci.*, vol. 63, no. 4, pp. 2217–2224, Aug. 2016.

[8] A. E. Wilson and M. Wirthlin, "Neutron radiation testing of fault tolerant RISC-V soft processor on Xilinx SRAM-based FPGAs," in *Proc. IEEE Space Comput. Conf. (SCC)*, Jul. 2019, pp. 25–32.

[9] A. B. de Oliveira *et al.*, "Evaluating soft core RISC-V processor in SRAM-based FPGA under radiation effects," *IEEE Trans. Nucl. Sci.*, vol. 67, no. 7, pp. 1503–1510, Jul. 2020.

[10] L. A. Aranda, A. Sanchez-Macian, and J. A. Maestro, "ACME: A tool to improve configuration memory fault injection in SRAM-based FPGAs," *IEEE Access*, vol. 7, pp. 128153–128161, 2019.

[11] H. Cho, "Impact of microarchitectural differences of RISC-V processor cores on soft error effects," *IEEE Access*, vol. 6, pp. 41302–41313, 2018.

[12] B. Du, S. Azimi, C. de Sio, L. Bozzoli, and L. Sterpone, "On the reliability of convolutional neural network implementation on SRAM-based FPGA," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst. (DFT)*, Oct. 2019, pp. 143–148.

[13] I. C. Lopes, F. Benevenuti, F. L. Kastensmidt, A. A. Susin, and P. Rech, "Reliability analysis on case-study traffic sign convolutional neural network on APSoC," in *Proc. IEEE 19th Latin-Amer. Test Symp. (LATS)*, Mar. 2018, pp. 7–12.

[14] F. Libano *et al.*, "Selective hardening for neural networks in FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 66, no. 1, pp. 216–222, Jan. 2019.

[15] 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 Trans. Nucl. Sci.*, vol. 68, no. 5, pp. 865–872, May 2021.

[16] H.-B. Wang, Y.-S. Wang, J.-H. Xiao, S.-L. Wang, and T.-J. Liang, "Impact of single-event upsets on convolutional neural networks in Xilinx Zynq FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 68, no. 4, pp. 394–401, Apr. 2021.

[17] C. Bolchini, L. Cassano, A. Mazzeo, and A. Miele, "Error modeling for image processing filters accelerated onto SRAM-based FPGAs," in *Proc. IEEE 26th Int. Symp. Test. Robust Syst. Design (IOLTS)*, Jul. 2020, pp. 140–145.

[18] L. A. Aranda, P. Reviriego, and J. A. Maestro, "Error detection technique for a median filter," *IEEE Trans. Nucl. Sci.*, vol. 64, no. 8, pp. 2219–2226, Aug. 2017.

[19] T. Tanaka, I. Ikeno, R. Tsuruoka, T. Kuchiba, W. Liao, and Y. Mitsuyama, "Development of autonomous driving system using programmable SoCs," in *Proc. Int. Conf. Field-Program. Technol. (ICFPT)*, Dec. 2019, pp. 453–456.

[20] A. Yamawaki and S. Serikawa, "A describing method of an image processing software in C for a high-level synthesis considering a function chaining," *IEICE Trans. Inf. Syst.*, vol. E101.D, no. 2, pp. 324–334, Feb. 2018.

[21] *Soft Error Mitigation Controller V4.1 LogiCORE IP Product Guide*, Xilinx, San Jose, CA, USA, PG036, Apr. 2018.

[22] C. E. Agüero *et al.*, "Inside the virtual robotics challenge: Simulating real-time robotic disaster response," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 2, pp. 494–506, Apr. 2015.

[23] Y. Sakemi *et al.*, "High intensity fast neutron beam facility at CYRIC," *Nucl. Technol.*, vol. 173, no. 2, pp. 210–217, Jul. 2014.

[24] Y. P. Chen, P. Maillard, M. Hart, J. Barton, J. Schmitz, and P. Kyu, "64 MeV proton single-event upset characterization of customer memory interface design on Xilinx XCKU040 FPGA," in *Proc. IEEE Radiat. Effects Data Workshop (REDW)*, Jul. 2017, pp. 144–147.

[25] R. Leveugle, A. Calvez, P. Maistri, and P. Vanhauwaert, "Statistical fault injection: Quantified error and confidence," in *Proc. Design, Autom. Test Eur. Conf. Exhib.*, Apr. 2009, pp. 502–506.

[26] S. Ramagond, S. Yellampalli, and C. Kanagasabapathi, "A review and analysis of communication logic between PL and PS in Zynq AP SoC," in *Proc. Int. Conf. Smart Technol. Smart Nation (SmartTechCon)*, Aug. 2017, pp. 946–951.

[27] S. T. Fleming and D. Thomas, "Injecting FPGA configuration faults in parallel," in *Proc. Int. Conf. Field-Program. Technol. (FPT)*, Dec. 2018, pp. 198–205.

[28] A. M. Keller and M. J. Wirthlin, "Single-event characterization of a Stratix 10 FPGA using neutron irradiation," in *Proc. IEEE Radiat. Effects Data Workshop*, Jul. 2019, pp. 173–178.