Alexandria Engineering Journal (2022) 61, 6181–6190

HOSTED BY

Alexandria University

Alexandria Engineering Journal

www.elsevier.com/locate/aej
www.sciencedirect.com

A new radiation-hardened architecture for holographic memory address calculation

Francisco Garcia-Herrero, Laura Rodríguez-Soriano *, Óscar Ruano, Juan Antonio Maestro

ARIES Research Center, Universidad Antonio de Nebrija, C/ Pirineos, 55, E-28040 Madrid, Spain

Received 25 March 2021; revised 17 August 2021; accepted 21 November 2021
Available online 02 December 2021

KEYWORDS

Error detection;
holographic data storage;
FPGA;
Memory protection

Abstract Architectures for the memory address calculation unit of a holographic memory device have been deeply analyzed in the literature. Finite-precision analysis and the selection of parameters to ensure a discrepancy between the hologram plane and the observation plane of less than 1% has been performed by other designers. However, a fault-tolerant architecture implemented on field-programmable gate array (FPGA) for an Optically Reconfigurable Gate Array has not been proposed yet. Due to the importance of this unit in environments with high levels of radiation, such as robots in nuclear disasters or satellites in space, which are the most common applications, there is a real interest in protecting this hardware. This work introduces two main contributions: i) replaces the calculation of distances through a Coordinate Rotation Digital Computer (CORDIC) unit in order to save area and ii) improves the ratio of fault-detection and extends the input range of the trigonometric computations to exploit more efficient error detection techniques. The derived architecture allows us to recover the FPGA from a faulty behavior in 95% of the cases, which avoids waiting for slower processes such as scrubbing. This real-time detection only requires an extra area of 13% with respect to the unprotected circuit and does not include any loss of precision compared to previous designs.

1. Introduction

The holographic data storage (HDS) systems have been investigated for more than two decades as an interesting alternative to digital memory systems for two reasons [1]: i) its storage density is larger than other devices; and ii) higher data transfer rates can be reached because of the use of non-binary information. However, the main problem of these systems is the proportion between signal-to-noise ratio (SNR) and bit-error rate (BER) for the optical part [2]. For years, proposals based on different optical modulation schemes have tried to improve the conditions of these storage resources [3,4]. Recently, this problem has been overcome, thanks to the addition of more advanced techniques such as convolutional neural networks (CNN) [5] that improve the optical noise tolerance during

* Corresponding author.
E-mail addresses: fgarciahe@nebrija.es (F. Garcia-Herrero), lrodriguezs5@alumnos.nebrija.es (L. Rodríguez-Soriano), oruano@nebrija.es (Ó. Ruano), jmaestro@nebrija.es (J.A. Maestro).
Peer review under responsibility of Faculty of Engineering, Alexandria University.

https://doi.org/10.1016/j.aej.2021.11.049

the demodulation [6]. The combination of CNN and error correction codes allow HDS to be good candidates for high-performance systems such as 8 K Super Hi-Vision [7,8].

On the other hand, a great effort to integrate HDS in high radiation environments has been done. One good example can be found in [9], where the objective was to be able to operate in similar conditions to the ones created in the accident of the Fukushima Daiichi nuclear power plant after the meltdown in March 2011. In this kind of environment only robots can operate, but with a limitation of about 10 h of work due to the radiation vulnerability. This vulnerability comes from the electronic parts of the system that, under these circumstances, lead to Single Event Upsets (SEUs), Multiple Bit Upsets (MBUs) or Single Event Functional Interrupts (SEFIs) [10] and these errors can result on a faulty performance. This makes it crucial to detect and repair them in a short period of time, so techniques like scrubbing for the configuration memory of the FPGA are not fast enough to detect and correct these errors before another SEU hits the designs [11].

One alternative to prevent radiation errors is to shield a radiation-hardened FPGA, but the total weight of the embedded system considerably increases. Whereas this is not a major problem in nuclear plants, in space applications it can be an issue.

Another possibility is to use full redundancy. Implementing in parallel more than two replicas of the architecture that wants to be protected, will allow us to detect and correct performance if one or two modules are damaged, just by comparing the output signals of the different replicas to check if all are behaving equally under the same inputs. Triple modular redundancy (TMR) is widely used in radiation environments. Five modular redundancy (5MR) and many modular redundancy (MMR) can also be implemented to increase the error mitigation rate. But the area and power cost increases, and the objective for space and battery-dependent applications (like robots) is to minimize the power consumption. Although N-Modular Redundancy is a common solution for ASIC implementations, FPGA designs allow other solutions based on detection that reduce the number of replicas needed to generate the trigger for the reconfiguration of the device when an error is detected.

In the last decade, one solution widely studied is the optically reconfigurable gate array (ORGAs), which has been designed to be radiation hardened in space applications [12,13]. However, all the holographic memory processing involved in the system is too power-consuming with optical logic, so the most efficient solution consists of developing the hungrier computational parts in FPGAs and endure them with ad hoc protections. In other words, to implement the operations that involve a higher number of calculations FPGA-based architectures will be applied to reduce power consumption, and at the same time, the design will consider fault-tolerance techniques to increase the percentage of detected errors in radioactive environments and activate reconfiguration in real-time to make the more computation reliable with a negligible cost.

In this paper, we present a memory address calculation unit for holographic memory devices implemented in an FPGA with full ad hoc protection. This unit is presented within a larger system such as ORGA but can be applied to any holographic memory devices as the calculations involved are the same. Results show that, compared to the modular redundancy methods, the increase in area is negligible and the efficiency in error detection is close to 95%, so scrubbing of the full configuration memory is necessary only in 5% of the cases.

This paper is structured as follows: in Section II the technique used to calculate holographic memory addresses is explained. In Section III an alternative radiation-hardened method is described. In Section IV the results are presented. The main conclusions of the paper are summarized in Section V.

2. Background

2.1. Optically Reconfigurable Gate Array (ORGA)

An ORGA is a type of multi-context field-programmable gate array composed of the holographic memory, a laser array and an ORGA-VLSI which contains an FPGA [14]. The block diagram of the ORGA system is included in Fig. 1 and described next.

The laser array (Fig. 1 A) is used to address the holographic memory contents. Although the information stored on the memory is robust, the semiconductor circuits of the laser are vulnerable to radiation, which can result in a wrong data read [14], so some method of protection is required to be implemented in (Fig. 1 B).

On the contrary, the holographic memory (Fig. 1 C) stores the configuration contexts of the programmable device. It is highly tolerant to radiation, due to its optical nature. Hence, the correct configuration data can be read even in high radiation conditions [11].

Finally, the last module is the ORGA-VLSI (Fig. 1 D), which consists of a programmable gate array with look-up tables (LUTs) and switching matrices. Its configuration is done optically, and it implies an optical bus between the holographic memory and the ORGA-VLSI.

2.2. Address calculation

The previously mentioned optical components of the holographic memory are extremely robust against radiation as they are manufactured with liquid cristal materials and prepolymer mixtures. In addition, the degradation suffered from the voltage-current ratio and the light power of the laser, due to the radiation, is also negligible [15]. However, the operations that are required to compute the addresses of the memory are implemented in CMOS technology that can be easily damaged by radiation.

The holographic memory address calculation is a heavy operation that involves the calculation of distances, trigonometric functions, additions, and products. This operation makes the conversion between the coordinates of the bright bits from the hologram plane (α, β) and the coordinates of those bits from the observation plane (x_i, y_i), Fig. 2.

To perform the conversion, the laser's intensity distribution read out from the hologram plane is calculated by the following equation: [1]

$$H(\alpha,\beta) = \sum_{i=1}^{B_N} cos\left(\frac{\pi}{\lambda L}\left\{(\alpha - x_i)^2 + (\beta - y_i)^2\right\}\right) \quad (1)$$

[1] We refer to [16] for more details on how to obtain this equation.

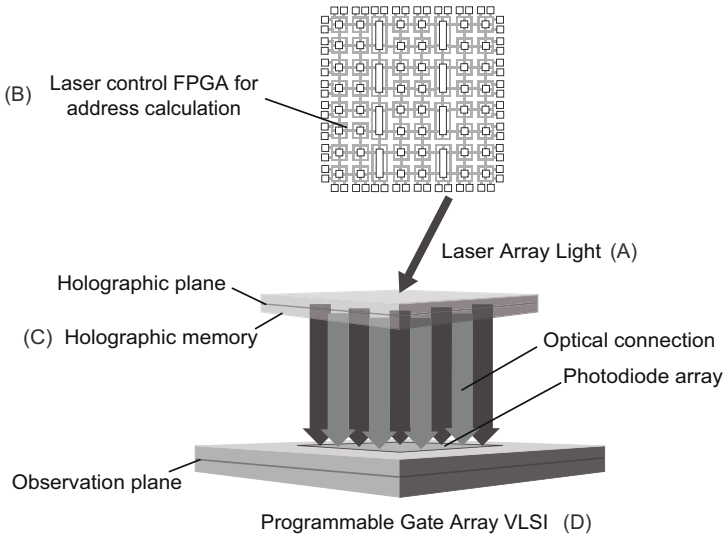

Fig. 1 ORGA block diagram.

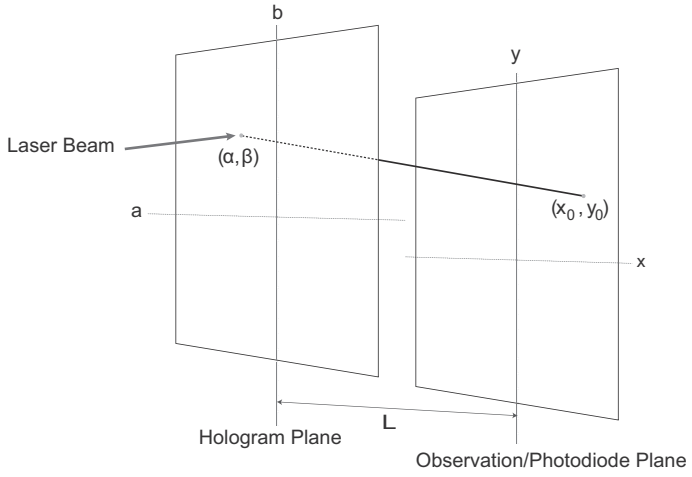

Fig. 2 Conversion of coordinates between the hologram plane and the observation/ photodiode plane.

where B_N is the number of bright bits in a configuration context; α and β are the hologram plane coordinates; x_i and y_i are the coordinates in the observation plane included in the photodiode array; λ is the wavelength of the laser and L is the gap between the hologram plane and the observation plane, Fig. 2.

2.3. Hardware architecture

Holographic memory calculation is a computationally demanding operation that will take too much time in general-purpose processors. For this reason, a custom hardware architecture must be applied to not reduce the high data transfer rates derived from HDS. Following this concept, a hardware FPGA accelerator that implements Eq. (1) is described in [16].

To map this calculation, the unit is divided into two subprocessors: i) the cosine and the distance calculator (Fig. 3) and and ii) the accumulator that adds the series of cosines (Fig. 4). As it can be seen in Fig. 4, the first module, that is the one with the highest cost, is replicated B_N times, so any improvement in terms of area or fault tolerance has a great impact on the final design, because the computation of the memory address is based on the accumulation of the B_N outputs of the cosine and distance calculation unit. Next, the implementation of each one of the units in [16] is described:

2.3.1. Cosine and distance calculation unit

$$cos\left(\frac{\pi}{\lambda L}\left\{(\alpha - x_i)^2 + (\beta - y_i)^2\right\}\right) \tag{2}$$

The computation of the cosines involved in Eq. (1) is also separated into two submodules: i) the calculation of the distances (which is the argument of the cosine) and ii) the cosine function.

For the first stage, the distances are computed using three adders-subtractors and four multipliers to process the squares and multiply by the constants. This is a straightforward solution that simplifies the pipeline distribution. However, it has

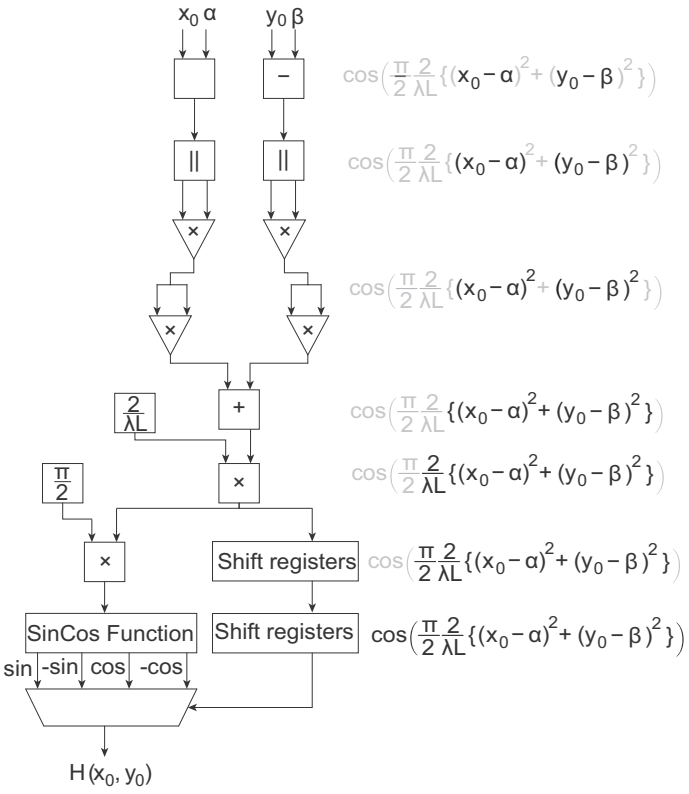

Fig. 3 Architecture of the cosine and distance calculation unit.

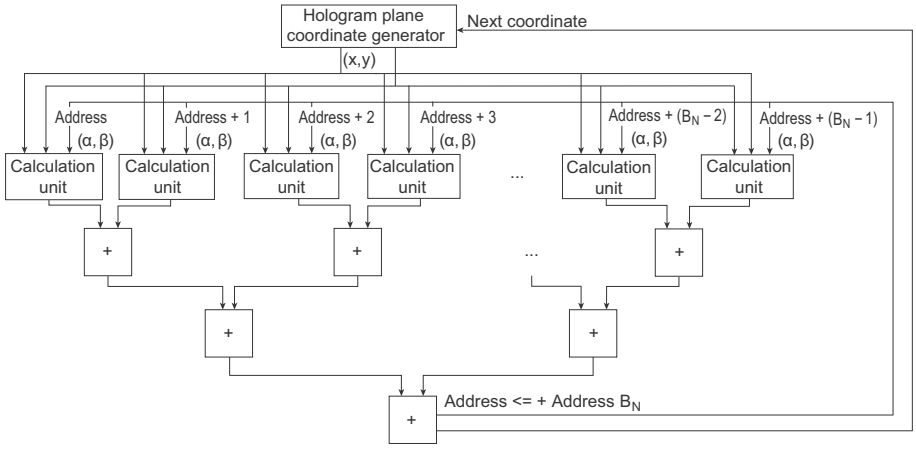

Fig. 4 Architecture of the accumulator unit.

a main drawback, which is the area and power consumption of the multipliers for the required precision. According to [16], 40-bit precision is necessary for the trigonometric function to make the HDS work properly. A more detailed study of the fixed-point analysis and a comparison with 64-bit floating-point processors is included in [16] showing an error of less than 0.39% using their parameters. In this work, we take as a starting point all these results and analysis and we also include in Fig. 5a-c a comparison between the patterns obtained with a 64-bit floating-point processor and a 40-bit finite precision FPGA architecture. These results show how the difference between both models is negligible by visual inspection. For this reason, the difference between both models was computed bit-wise to check that the error was similar to white noise and the difference was less than 0.39%. This means that we compared the outputs of the 64-bit implementation and 40-bit implementation, using the same input signals, and we showed that the difference at the output was smaller than 0.39% and the distribution of the error generated when the precision was reduced was a Gaussian distribution, which is the same as the one generated by white noise as can be seen in Fig.5c.

For the second part, the cosine function is calculated through a Coordinate Rotation Digital Computer (CORDIC)

(a) (b) (c)

Fig. 5 Diffraction patterns plotted with Matlab and processed with: a) Intel Core i7-6700HQ @2.6 GHz, b) Nexys4 DDR (xc7a100tcsg324-1) FPGA board and c) Difference between patterns computed with Matlab. $\lambda = 532$ nm, $L = 10$ mm.

IP Core, two shift registers and a multiplexor to select the output between $sin, -sin, cos$ and $-cos$. These shift registers and the multiplexors are necessary because the CORDIC input range is from 0 to $\pi/2$, which implies adding all this extra logic to calculate the result in the range $-\pi$ to π. Although the use of the core is convenient, a comparison with a full range CORDIC is not performed.

As a final remark, none of these parts are protected to detect a failure, so they are susceptible to have a malfunction.

2.3.2. Accumulator

For a high-speed design, the previous calculation unit needs to be implemented B_N times (B_N units in parallel). For the specific example of the ORGA in [16], which is the one that is protected in this work, $B_N = 8$ is selected to ensure that the values of the pixels in the hologram plane and the ones read in the observation plane are almost identical. Considering also all the finite precision values of the rest of the architecture, the error between both planes is less than 0.39%. The outputs are connected to an accumulator, based on a tree-adder, to calculate each memory address. These results show how the accumulator part has a negligible area compared to the cosine calculation units, being the first one only 8 adders and the second one 24 adders-subtractors, 32 multipliers, 8 CORDIC IP Cores, 16 shift registers, 8 multiplexors and an accumulator with 40-bit precision. This first analysis helps us to understand the importance of the protection of the cosine unit if it is going to work on a strong radiation environment, as it is the largest part, in terms of area, and hence, the most vulnerable ones in the FPGA design.

2.4. Existing fault-tolerant architectures for CORDIC

CORDIC unit is a fundamental processor for the holographic memory address calculation. For this reason, it is mandatory to protect it against radiation to ensure the correct behavior of the storage system. In that sense, several techniques have been published in the literature, based on data coding [17], hardware redundancy [18], and time redundancy [19]. Each of these solutions provides different advantages depending on the constraints selected for the design. For example, the use of error correction codes gives a satisfactory detection capacity with single faults that induce single errors on the output with very low redundancy [17]. However, the CORDIC unit has a peculiarity, if a single fault is produced in an intermediate stage of its pipelined architecture, the error is propagated towards the output, affecting more than one bit. This makes the use of the previous codes [17] impractical when this type of faults hit the design. To overcome this situation some selective techniques as the ones introduced in [18] can be applied. This solution checks the intermediate results of the CORDIC avoiding the undetection of multiple errors in the output produced by a single hit. The problem with these solutions is that they do not only introduce overhead in the area, but also in latency, so they are not recommended for applications in which the time to detect a fault is critical. The last type of solution is based on redundancy in area or time. For this kind of solution, the most common architectures are the ones based on TMR. As it is stated in the introduction, these solutions are efficient for ASIC implementations, where reprogramming is not an option. For FPGA implementations, which is the case of this work, more than two replicas are considered inefficient. In the next section, a customized protection for the CORDIC units involved in the holographic memory address calculation is described, taking into consideration i) the error propagation in the pipeline architectures, ii) the latency constraints, and iii) the FPGA implementation.

3. Protected unit

In this section, a protected architecture for the holographic memory address calculation is introduced. The objective of this new design is to try to minimize the area increase of the protected design but, at the same time, improve the percentage of errors detected in order to start a reconfiguration of the FPGA without waiting for the scrubbing process to end. To make comparisons easier we split this section into the two main parts of the cosine calculation unit (Section II-C): i) the calculation of the distances and ii) the cosine function. In the first module, the strategy that has been followed is to exchange the straightforward distance calculation unit by a CORDIC configured in vector mode. This modification allows us to exploit repeated computation and to build a DMR-like architecture. In the second module, an algorithmic approach has been chosen, using the $sin^2 + cos^2 = 1$ property to achieve the protection.

In the following, the details of these approaches will be presented, describing how they benefit the area overhead. All the experiments described in the next sections were performed with Vivado, System Generator, ModelSim and Matlab. The derived hardware architectures were implemented on a Nexys4 DDR (xc7a100tcsg324-1) FPGA board and verified comparing the outputs with the golden model architecture from [16].

3.1. Calculation of the distances

In Fig. 3, the two multipliers that are used for the square computations are extremely area demanding for the 40-bit precision system. Implemented on the selected FPGA each one of these multipliers occupies 1789 look-up tables (LUTs). In addition to this, there is a module operation along with the subtraction and addition involved in computing the distances. As an alternative, one extended solution in communication systems is to calculate the distances between two points with the use of the CORDIC algorithm in the vectorization mode (Fig. 6).

As it can be seen, by means of implementing two CORDIC modules and multiplying its outputs and each one of them by the constant $\sqrt{\frac{\pi}{\lambda L}}$ we can obtain the argument of the cosine that we are looking for $\left(\frac{\pi}{\lambda L}\left\{(\alpha - x_i)^2 + (\beta - y_i)^2\right\}\right)$ with a series of advantages, Fig. 7.

The first advantage is that for the same precision, the processing of the distance with the two CORDIC units is 3297 LUTs, which is slightly less than the area of the two original multipliers, which is 3578 LUTs. The second advantage is that, as can be seen in Fig. 7, both CORDIC compute the same output, so they become a natural DMR circuit, detecting errors between them and using that signal to reconfigure the FPGA in case of error. This protection cannot be included in the original multiplier design as each output has a completely different value in absence of error. Finally, the multiplication of both outputs is greatly simplified, if we assume that both inputs are equal. The logic of the multipliers is reduced to 614 LUTs. If DMR is applied for the output of the multipliers, 1228 LUTs are spent. Note that if the inputs are not equal the simplified multipliers are not useful, but the previous DMR signal will detect the misfunction and correct the error via reconfiguration. In addition, as it is detailed in [20] this CORDIC architecture is fully pipelined, allowing very high maximum frequency.

To sum up this subsection, the full calculation of distances unit based on multipliers requires: 4647 LUTs (with all the logic from Fig. 3) and the new proposal requires 4776 LUTs (with all the DMR protection from Fig. 7). So, with a cost of 2.8% more of area this part of the unit is protected against errors in a high percentage and obtains almost the same result with a deviation of less than 4e-6 in the worst case, as we will see in the next section.

3.2. Cosine function

For this module, a CORDIC is also applied but with a different mode of operation (Fig. 8). The aim of this part is to compute the cosine value. Although the use of the Altera's CORDIC IP in [16] allows very fast prototyping, it can be checked that the difference in terms of area between the CORDIC with input ranges from 0 to $\pi/2$ (with all the extra logic involved) and one implementation, like the one from [20], with a range between $-\pi$ to π, have very similar area results. This was verified with both Altera and Xilinx IDEs showing a difference of fewer than 100 LUTs.

As happened with the previous unit, implementing this architecture allows a more efficient way to protect the design and detect possible errors. Implementing the full-range CORDIC, allows us to use simple trigonometric properties to detect if there is an error on the CORDIC module. As the CORDIC always provides the sine/cosine functions as output, Eq. (3) can be applied to detect faults in this part of the process [21].

$$\cos^2(2\pi \cdot P \cdot f) + \sin^2(2\pi \cdot P \cdot f) = 1 \tag{3}$$

On the contrary, the original architecture can protect only the behavior of the CORDIC itself, but it cannot detect errors on the rest of the logic required to cover the range out of 0 to $\pi/2$. In Fig. 9 it is highlighted the part of the circuit that should require a technique such as DMR, apart from the implementation of Eq. (3) to fully protect this area of the design. So, it is trivial to see how using the reduced range CORDIC will increase the area resources needed for the protection of this unit.

This trigonometric property can be implemented by adding a circuit that calculates the square of CORDIC outputs (both sine and cosine) and verifies the equality, Fig. 10. Since the CORDIC has a finite precision, this result will suffer an oscillation (γ). For this reason, the result of this trigonometric calculation must be in the range from $1-\gamma$ to $1+\gamma$ to verify the property and to assume that no errors have occurred [21]. The range is determined by simulation and depends on the precision of the CORDIC. The unprotected version of this submodule has 1249 LUTs and the protected one has 1891 LUTs, while a DMR is twice the unprotected version, about 2500 LUTs. So, the increase of this part is 51% compared to the unprotected one.

4. Results

When the previous architectures are implemented on a Nexys4 DDR FPGA for the complete holographic memory address calculation unit, with $B_N = 8$ results are the following (Table 1).

4.1. Area results

The area increment compared to the unprotected version is 13.1%. Here it is important to highlight how the calculation of distances requires more area resources than the cosine calculation and hence, the efficient protection technique applied here has a greater impact when the different units are parallelized. On the other hand, 50% of the extra area required by the cosine unit has a negligible impact on the increment of the final design. Compared to a fully DMR, the architecture has a 43.4% less area. It is important to remark that the power consumption is also proportional to the area of the design.[2]

4.2. Maximum frequency and power consumption

Both designs reach the same speed (Table 2) as the one proposed in this work has distributed the pipeline registers to obtain the same maximum clock frequency. However, the new architecture introduced here has a considerable reduction of power consumption due to the simplification in the number of power hunger operations, such as multipliers.

[2] Note that Fig. 7 and Fig. 10 are implemented in parallel, so there is no sharing of CORDIC between them as they implement different functionalities of the algorithm. This can be seen in Fig. 12 where the full architecture is included.

Fig. 6 CORDIC configuration for distance calculation [20].

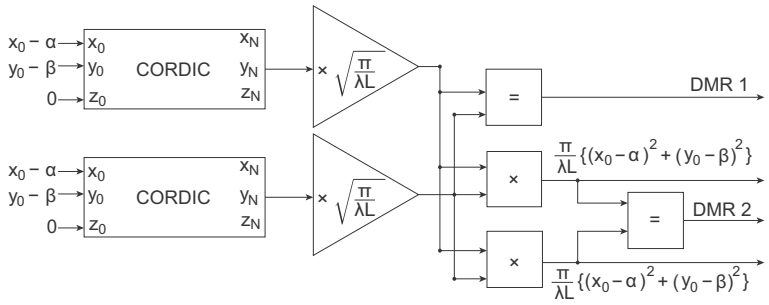

Fig. 7 Alternative protected calculation of distance unit.

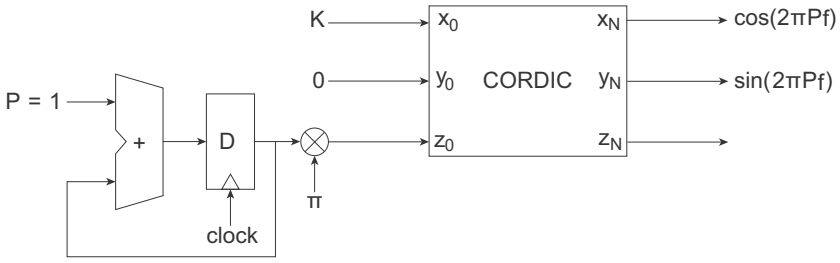

Fig. 8 CORDIC configuration for sine and cosine calculation.

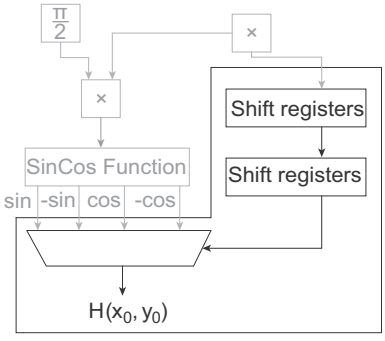

Fig. 9 Highlighted is the part of the design that cannot be protected with Eq. (3).

4.3. Fault tolerance analysis

To characterize the behavior of the different proposals we used the tool in [22] with the setup in Fig. 11. This tool, called ACME [15], injects errors in real-time in the configuration memory of the FPGA to check the behavior at the output emulating a bit-flip caused by radiation. To do this, the essential bits of the design under test are extracted to know which bits of the configuration memory are part of the architecture that we want to model. Once the essential bits are extracted, the tool injects in every single bit of the design. To inject errors in the configuration memory, it is possible to use the Xilinx Soft Error Mitigation (SEM) IP Controller [23]. The tool can be downloaded in [24] to be tested in any other design. For the different experiments, one ROM with the inputs feeds

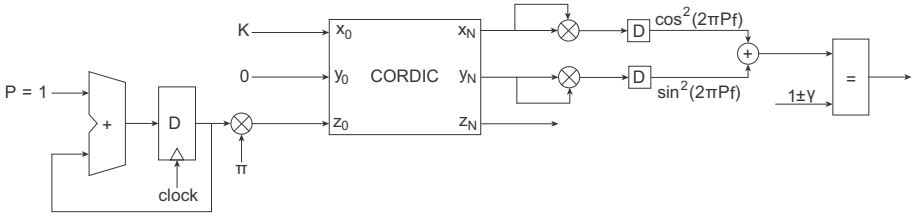

Fig. 10 Protected cosine function.

Table 1 Area overhead and detection rate of the different unprotected and protected architectures.

Architecture	LUTs	Error Detection	Overhead
Single unprotected distance calculation unit [16]	4647	N/A	0
Single protected distance calculation unit (this work)	4776	99.7%	2.8%
Single unprotected cosine unit [16]	1249	N/A	0
Single protected cosine unit (this work)	1891	84.7%	51%
Unprotected version of the full architecture, $B_N = 8\times$ [16]	47168	N/A	0
DMR of the unprotected full architecture, $B_N = 8\times$ [16]	94430	99.7%	100.2%
Protected version of the full architecture (this work), $B_N = 8$	53336	95%	13.1%

Table 2 Maximum clock frequency and power consumption of the protected architectures

Architecture	Maximum clock frequency	Power consumption
DMR of the unprotected full architecture [16], $B_N = 8$	143 MHz	6 W
Protected version of the full architecture (this work), $B_N = 8$	147 MHz	0.32 W

the golden model (CIRCUIT 1 in the figure) and the design that will suffer the error injection (CIRCUIT 2). Then with a checker (which is a comparator and a counter), we check if the outputs of both circuits are equal. If they are different, the counter of the number of errors is increased. It is important to remark that not all the injections cause an error in the output, some errors are masked because of the architecture itself. For our experiments, CIRCUIT 1 is the unit without any error (golden model) and CIRCUIT 2 is the unit that suffers the error injection. This error injection is exhaustive, all the essential bits of CIRCUIT 2 have an error injection. After each injection, we check the behavior of the design during some samples, if one or more errors are detected during that period the counter of errors is increased and we inject in the following region of the design, and so on. In parallel, the signal of error detection, that activates the reconfiguration of the FPGA in presence of error, is registered. Comparing the number of cases in which the error-detection signal is activated to the number of errors with respect to the golden model, the detection ratio is computed.

These experiments showed that for the detection of errors, the proposal for the calculation of distances has a 99.7% of success, while the cosine function has 84.7% of detection. As happens with the area, the total number of possible errors that can be generated in the distance module is larger than the number of errors that can be generated in the cosine function, so when we analyze the full design with the B_N replicated modules the rate of success is of more than 95%. With this, we can sum up that only in 5% of the cases the error detection will not active the reconfiguration of the FPGA and, hence, it will be necessary to wait for the scrubbing, but in 95% of the cases we will detect, reconfigure and program the FPGA automatically at a cost of 13.1% of the area. Compared to the fully DMR, this solution seems more efficient as it does not require an increase of 100% of the area and only loses a 4.7% of error detection.

Other traditional solutions such as error-correcting codes have not been considered as they have two disadvantages for this application: i) the delay in terms of latency introduced by the computation and comparison of 40-bit parity-check equations is higher than the one introduced by the comparators, reducing the maximum frequency that is achieved to reprogram the FPGA in case of error; ii) the number of parity bits that need to be added to achieve a level of protection of more than 90% is higher than 100%, assuming the most efficient codes in terms of detection which increases considerably the area required and without any benefit in terms of speed.

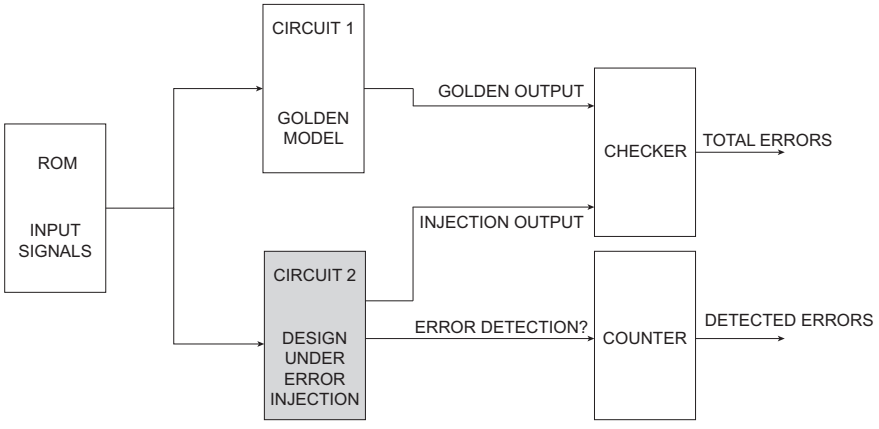

Fig. 11 Setup of the error injection experiments.

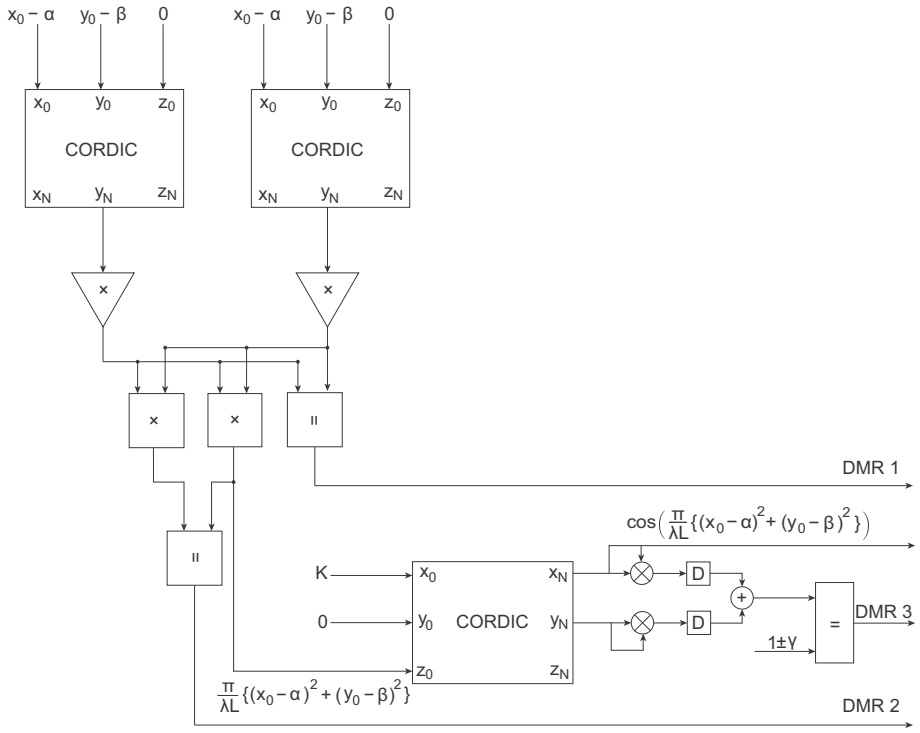

Fig. 12 Protected version of the full architecture.

5. Conclusion

In this paper, a fault tolerance architecture for the memory address calculation unit of a holographic memory device is described. The architecture is implemented on FPGA for an Optically Reconfigurable Gate Array, which is designed to work in environments with high levels of radiation, such as robots in nuclear disasters or satellites in space. The solution provided in this work replaces the calculation of distances through a CORDIC unit in order to save area and improve the ratio between the detected faults and the total number of faults, and extends the input range of the trigonometric computations to exploit more efficient error detection techniques. The derived architecture allows us to recover the FPGA from a faulty behavior in 95% of the cases, which avoids waiting for more slow processes such as scrubbing. This real-time detection only requires an extra area of 13% compared to the unprotected design, saving 87% of the resources compared to the traditional protected solution, DMR, at a cost of not detecting 5% of the errors on the fly. The only limitation of this architecture is that for ASIC implementation other fault-tolerant solutions should be explored, as the reprogramming of the memory configuration is not an option.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] J.F. Heanue, M.C. Bashaw, L. Hesselink, Volume Holographic Storage and Retrieval of Digital Data, Science 265 (5173) (1994) 749–752, https://doi.org/10.1126/science.265.5173.749, https://science.sciencemag.org/content/265/5173/749.full.pdf, https://science.sciencemag.org/content/265/5173/749.

[2] D.P.H.J. Coufal, G.T. Sincerbox, Holographic data storage, Springer, New York, 2000.

[3] Euiseok Hwang, Jaewoo Roh, Kunyul Kim, Byongbok Kang, Jooyoun Park, Heungsang Jung, Code-word complementing block modulation code for holographic data storage, in: International Symposium on Optical Memory and Optical Data Storage Topical Meeting, 2002, pp. 195–197.

[4] C.D. Nguyen, N. Xuan Pham, C.C. Duong, L. Cong Nguyen, Multilevel modulation coding for four-level holographic data storage systems, in: 2020 International Conference on Advanced Technologies for Communications (ATC), 2020, pp. 30–34, https://doi.org/10.1109/ATC50776.2020.9255459.

[5] T. Shimobaba, Y. Yamamoto, I. Hoshi, T. Kakue, T. Ito, Data page classification in holographic memory using binary neural network, in: 2020 IEEE 18th International Conference on Industrial Informatics (INDIN), vol. 1, 2020, pp. 511–514. https://doi.org/10.1109/INDIN45582.2020.9442176.

[6] Y. Katano, T. Muroi, N. Kinoshita, N. Ishii, N. Hayashi, Data demodulation using convolutional neural networks for holographic data storage, Jpn. J. Appl. Phys. 57 (9S1) (2018) 09SC01, https://doi.org/10.7567/jjap.57.09sc01.

[7] Y. Katano, T. Nobukawa, T. Muroi, N. Kinoshita, N. Ishii, Effective Data-Decoding Method by Combining Convolutional Neural Network and Spatially Coupled Low-Density Parity-Check Code for Holographic Data Storage, in: ISOM'19 Tecnical Digest, Niigata, Japan, 2019.

[8] Y. Katano, T. Nobukawa, T. Muroi, N. Kinoshita, N. Ishii, Efficient decoding method for holographic data storage combining convolutional neural network and spatially coupled low-density parity-check code, ITE Trans. Media Technol. Appl. 9 (3) (2021) 161–168, https://doi.org/10.3169/mta.9.161.

[9] T. Fujimori, M. Watanabe, A 400 Mrad radiation-hardened optoelectronic embedded system with a silver-halide holographic memory, in: 2018 NASA/ESA Conference on Adaptive Hardware and Systems (AHS), 2018, pp. 218–224.

[10] R. Baumann, Soft errors in advanced computer systems, IEEE Des. Test Comput. 22 (3) (2005) 258–266.

[11] M. Watanabe, T. Fujimori, Holographic scrubbing technique for a programmable gate array, in: 2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS), 2015, pp. 1–5.

[12] T. Watanabe, M. Watanabe, Robust holographic storage system design, Opt. Express 19 (24) (2011) 24147–24158, https://doi.org/10.1364/OE.19.024147, http://www.opticsexpress.org/abstract.cfm?URI=oe-19-24-24147.

[13] A. Ogiwara, M. Toda, J. Ishido, M. Watanabe, H. Kakiuchida, Effects of a radiation dose in gamma-ray irradiation fields on holographic gratings formed by liquid crystal composites, OSA Continuum 4 (2) (2021) 514–528, https://doi.org/10.1364/OSAC.415702, http://www.osapublishing.org/osac/abstract.cfm?URI=osac-4-2-514.

[14] S. Fujisaki, T. Fujimori, M. Watanabe, An optically reconfigurable gate array workable under a strong gamma radiation environment, in: 2019 IEEE Workshop on Microelectronics and Electron Devices (WMED), 2019, pp. 1–4.

[15] K. Sano, Applied Reconfigurable Computing, Springer International Publishing, 2015.

[16] T. Fujimori, M. Watanabe, Holographic Memory Calculation FPGA Accelerator for Optically Reconfigurable Gate Arrays, in: 2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech), 2017, pp. 620–625.

[17] T.R.N. Rao, Error Coding for Arithmetic Processors, Academic Press, Inc., USA, 1974.

[18] J. Kwak, V. Piuri, E.E. Swartzlander, Fault-tolerant high-performance CORDIC processors, in: Proceedings IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, 2000, pp. 164–172, https://doi.org/10.1109/DFTVS.2000.887154.

[19] Jae-Hyuck Kwak V. Piuri, E.E. Swartzander, Time-shared TMR for fault-tolerant CORDIC processors, in: 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221), vol. 2, 2001, pp. 1241–1244. https://doi.org/10.1109/ICASSP.2001.941149.

[20] P.K. Meher, J. Valls, T. Juang, K. Sridharan, K. Maharatna, 50 Years of CORDIC: Algorithms, Architectures, and Applications, IEEE Trans. Circuits Syst. I Regul. Pap. 56 (9) (2009) 1893–1907.

[21] L.A. Aranda, F. Garcia-Herrero, L. Esteban, A. Sanchez-Macian, J.A. Maestro, Radiation Hardened Digital Direct Synthesizer With CORDIC for Spaceborne Applications, IEEE Access 8 (2020) 83167–83176.

[22] L.A. Aranda, A. Sanchez-Macian, J.A. Maestro, ACME: A Tool to Improve Configuration Memory Fault Injection in SRAM-Based FPGAs, IEEE Access 7 (2019) 128153–128161.

[23] C.U. Xilinx, San Jose, Soft Error Mitigation Controller LogiCORE IP Product Guide (PG036), sep. 2015.

[24] ACME Tool repository, http://www.nebrija.es/aries/acme.htm (accessed: 2020-12-12).