INTEGRATION, the VLSI journal 42 (2009) 128– 136

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

Protection against soft errors in the space environment: A finite impulse response (FIR) filter case study

J.A. Maestro [a,*], P. Reviriego [b], P. Reyes [a], O. Ruano [a]

[a] Universidad Antonio de Nebrija, Madrid, Spain
[b] Universidad Carlos III de Madrid, Madrid, Spain

ARTICLE INFO

Article history:
Received 3 February 2007
Received in revised form
28 March 2008
Accepted 4 April 2008

Keywords:
Fault tolerance
Soft errors
Radiation
Error detection and correction codes
Digital filters

ABSTRACT

The problem of radiation is a key issue in Space applications, since it produces several negative effects on digital circuits. Considering the high reliability expected in these systems, many techniques have been proposed to mitigate these effects. However, traditional protection techniques against soft errors, like Triple Modular Redundancy (TMR) or EDAC codes (for example Hamming), normally result in a significant area and power overhead. In this paper we propose a specific technique to protect digital finite impulse response (FIR) filters applying the "system knowledge". This means to study and use the singularities in their structure in order to provide effective protection with minimal area and power. The results obtained in the experimental process have been compared with the protection offered by TMR and Hamming codes, in order to prove the quality of the proposed solution.

1. Introduction

Space is an environment rich in radiation and charged particles. The main radiation sources that can be found in this environment are the solar wind (mainly protons) and cosmic radiation consisting of heavier particles. When these particles interact with the Earth's atmosphere, other kinds of phenomena are produced, as neutrons and gamma-ray photons. These are also radiation sources that can affect both digital devices in low-orbiting satellites and on-ground level electronic systems.

There are several studies [1] reporting problems on systems implemented in space missions, which range from soft errors to the total damage of the circuit. This has motivated several research lines trying to find solutions to these problems both at the technological level and at the design level. These efforts are driven by researchers in the industry, the academia and also in the different space agencies.

In details, when a radiation particle strikes on a semiconductor device and it goes through the electric field region, it generates a large number of electron–hole pairs. If such an event occurs near the depletion region of a reverse biased p–n junction, the free carriers are efficiently collected, increasing the electric field across the mentioned junction. This generates a current flowing through the device that causes a transient pulse. When this transient pulse, known as Single Event Effect (SEE), occurs in or is registered by a storage element, it causes a functional or data result error in the circuit which is referred to as Single Event Upset (SEU) or bit flip [2,3]. Otherwise, it is referred to as a Single Event Transient (SET).

Moreover, as the microelectronic industry technology processes reduce the size of the devices, lower operation voltages are needed, and the reduction of the charge stored on the circuit nodes increases the failure rate of the semiconductor devices due to soft errors. The main reason for this is that under these circumstances, even low energy particles (which have a greater frequency of occurrence than the high energy ones) can cause upsets.

Many application fields are affected by these phenomena, especially those where radiation is strongly present. The Space field, which has been the focus of this paper, is especially interesting due to its inherent constraints on performance, area and power [4]. Traditionally, most soft errors occur in memory arrays, DRAM and SRAM cells, but with the reduction of device sizes, Single Event Transient (SET) events are increasingly important as noted in [5,6] for a 45 nm technology. Another important area affected by soft errors in Space are the communication systems. Most of the devices that operate in Space (e.g. satellites) need to perform a lot of communication operations, some of them critical. As digital filters are widely used in communication systems, the development of specific protection techniques for those filters would have a direct benefit in many

[*] Corresponding author. Tel.: +34 914521100; fax: +34 914521110.
E-mail addresses: Jmaestro@nebrija.es (J.A. Maestro), revirieg@it.uc3m.es (P. Reviriego), Preyes@nebrija.es (P. Reyes), oruano@nebrija.es (O. Ruano).

0167-9260/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.vlsi.2008.04.002

space applications. In this paper, the presented case study is devoted to finite impulse response (FIR) filters, as they exhibit a regular structure suitable for specific protection techniques and are commonly used both in communications and signal processing applications.

To mitigate the effects of soft errors, a number of techniques can be used at the physical level (device size and structure) [7]. In addition to those techniques, redundancy can be introduced in the design so that it can detect and correct soft errors [8]. To deal with SEUs, a common approach is Triple Modular Redundancy (TMR), which triplicates the flip-flops in the design and adds logic to vote in case of conflict. If SETs are also to be considered, Functional Triple Modular Redundancy (FTMR, which also triplicates the combinational logic) can be used [8], although it was originally designed to provide protection against SEU-induced alteration of combinational logic in SRAM-based reprogrammable FPGAs.

Other general techniques to deal with SEUs by introducing redundancy are Error Detection and Correction (EDAC), like Hamming codes, where one encoder, one decoder and several additional registers to store redundancy are introduced in each register. The details of these techniques will be explained in detail in the following section.

On the contrary, the approach to deal with SEUs presented in this paper is based on applying circuit specific techniques that exploit the inherent redundancy or fault tolerance of some circuits [9,10], what we call *to apply the system knowledge*. The advantage of this is the production of custom-tailored solutions for each family of circuits, with good protection levels and a quasi-optimal implementation, something that general techniques like TMR or Hamming coding cannot achieve.

Objectives: In this paper, we propose an approach to protect FIR filters against SEUs. The proposed approach will be put in perspective with other existing solutions, uncovering some weaknesses associated to these schemes. Afterwards, we will evaluate the proposed technique using a soft-error simulation platform implemented by the European Space Agency [11,12]. Finally, the different approaches will be compared in terms of protection effectiveness, impact on the maximum operating speed of the circuit and area as figure of merit for complexity.

2. Related work

In this section, some work related to the topics of this paper will be presented. The problem of radiation on electronic devices has been traditionally addressed in literature. One of the major concerns has always been how soft errors are induced, from a physical point of view. This implies studying not only the source of errors, but also modeling the event arrival rate through probabilistic distributions that can be used to foresee the behavior of the circuit. This leads to the following question, once errors have been predicted: how to make the circuit fault tolerant. There are different approaches in literature, most of them based on redundancy (duplication and mainly triplication). These different techniques need to deal with the extra hardware cost associated with a higher fault tolerance. Finding an appropriate method to detect weaker areas to be protected, adding only a minimal hardware overhead, is a very recurring research line. In the following paragraphs, some references about these topics are provided:

A classic reference by Ziegler is offered in [13], where the basic physics of radiation effects is detailed. Different rates of errors at several terrestrial positions are described, providing a quantitative analysis of the radiation effects.

A reference that deals with a similar problem to the one stated in this paper is [14]. A formal solution is proposed in order to detect errors in linear digital state variable systems. Using a tool called the *gain matrix*, the error propagation along the circuit paths is analyzed. This allows studying weaker areas in the circuits that should be protected. Although the results are promising, nothing is said about the implementation cost of the solution.

One of the factors that measure the sensitivity of circuits to radiation is the error rate, which is defined as the number of error events that occur in the circuit in a time unit. Several works try to provide models for this error rate, in order to foresee the behavior of the circuit in a particular environment. A Soft Error Rate computation algorithm is presented in [15], which can be applied to combinational circuits. The parametric waveform model is based on the Weibull function, which represents the distribution of article strikes at the different nodes. Experiments show that the algorithm is linear in the number of nodes, and results are close to SPICE simulations.

A methodology to compute the effects of charged particles inducing delay errors (Soft Delay Errors) is presented in [16]. A soft delay is a kind of temporary error produced by high energy particles striking on CMOS combinational circuits. This produces a slower operation of the element, what usually leads to a loss of synchronization with the rest of the circuit, and therefore to a soft error at the output. In order to overcome this problem, the concept of “node sensitivity” is defined and computed to employ node hardening techniques, and therefore, increase the reliability of CMOS circuits.

Techniques to detect and correct errors are very common too. The goal of such techniques is to mitigate the effects of radiation, both by detecting errors when they happen and by trying to correct them, thus getting rid of their negative effect. In [17], the problem of Concurrent Error Detection (CED) is discussed in Burst-Mode machines. An enhanced duplication process is proposed in order to give a solution to this problem, showing an interesting saving in hardware.

A technique to minimize the impact of soft errors in circuits is presented in [18]. Through the use of complementary pass transistor devices, the gates affected by SEUs are isolated, and therefore their negative effect is removed. This is achieved with limited area, delay and power overheads.

In [19], the problem of sub-65 nm designs is described. This kind of technology needs built-in logic for error protection. Since it is stated that classical fault-tolerance techniques for soft error detection are expensive, a recently developed Built-In-Soft-Error-Resilience (BISER) technique is proposed, which seems effective for soft error blocking or detection.

3. Conventional soft error protection techniques

Traditional techniques like Triple Modular Redundancy and Error Detection and Correction codes are usually employed to deal with SEUs in several application fields, like avionics, space and medical areas. This section describes those conventional techniques.

3.1. Triple modular redundancy

Triple Modular Redundancy, TMR, enhances the fault tolerance of the target circuit by triplicating the storage elements (registers, flip-flops, etc.) and adding a voting logic that selects the majority value of each set of replicated storage units (see Fig. 1).

Using TMR, the highest number of SEUs that the protected circuit can support in an n-bit register without errors in its behavior would be n, providing that each SEU occurs in different bits of the register.

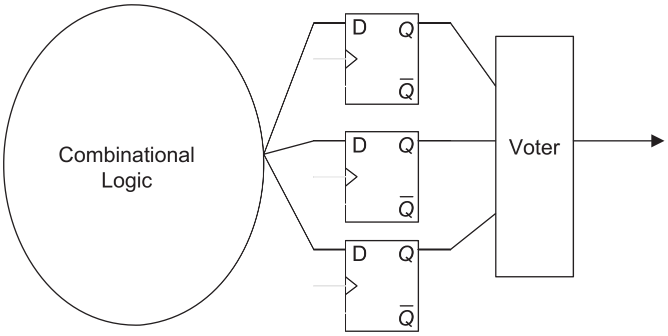

Fig. 1. Illustration of TMR.

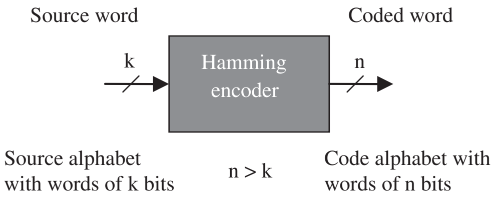
Source word
Coded word
k
Hamming encoder
n
Source alphabet with words of k bits
n > k
Code alphabet with words of n bits

Fig. 2. Hamming Encoder.

If two simultaneous SEUs occur on the same set of registers used to store and protect a single bit, the error is propagated and it can cause a functional/data failure.

If it is necessary to deal with SETs, the use of Functional TMR, FTMR, which also triplicates the combinational logic, must be considered.

3.2. Hamming codes

Similarly, error detection and correction codes, sometimes referred as parity codes, could be used in place of TMR in order to protect the circuit against SEUs. One example are Hamming EDAC codes [20], which are named as *Hamming* (n,k) where n represents the number of bits of the coded word (the word with parity and data bits) and k is the number of data bits of the initial word, as it is illustrated in Fig. 2.

The main properties of Hamming codes are that they can detect double errors and correct single errors (SEC-DED). The correction and detection capabilities are determined by different properties of the code, as the Hamming distance.

Hamming encoding is performed by using combinational logic needed to calculate the parity bits of each source word to be coded. This logic is determined by the code generator matrix, G, a matrix that indicates how to calculate parity bits from the data bits. In the case of a *Hamming* (12, 8) code (which is the one used in this paper), a possible G matrix would be

$$G = \begin{bmatrix} 1&0&0&0&0&0&0&0&1&1&0&0 \\ 0&1&0&0&0&0&0&0&1&0&1&0 \\ 0&0&1&0&0&0&0&0&0&1&1&0 \\ 0&0&0&1&0&0&0&0&1&1&1&0 \\ 0&0&0&0&1&0&0&0&1&0&0&1 \\ 0&0&0&0&0&1&0&0&0&1&0&1 \\ 0&0&0&0&0&0&1&0&1&1&0&1 \\ 0&0&0&0&0&0&0&1&0&0&1&1 \end{bmatrix} \tag{1}$$

In this way, the coded word will be obtained multiplying (in a binary way) the source word and G.

$$c = x \cdot G \tag{2}$$

where G is the generator matrix, x the word to be coded and c is the final coded word. This coded word represents its data bits in the first k bits and the parity bits in the last $n-k$ bits.

In order to make Hamming codes operative, from a protection point of view, some hardware needs to be added in order to implement the error detection process and the error correction mechanism.

The error detection process is performed by the study of the *error syndrome*, *s,* which is a standard concept that is calculated as follows:

$$s = c \cdot H^{\mathrm{T}} = \left\{ \begin{matrix} 0 & \text{No errors} \\ \neq 0 & \text{Errors} \end{matrix} \right\} \tag{3}$$

where c is the coded word and H^T is called the *check parity matrix* (a matrix that is orthogonal to G). In this way, it can be proved that for every coded word without error, a null syndrome (i.e. equal to zero) would be obtained. On the other hand, if an error occurs in the coded word, the obtained syndrome will not be zero.

The H^{T} matrix dual to the G matrix proposed in Eq. (1) is shown in Eq. (4):

$$H^{\mathrm{T}} = \begin{bmatrix} 1&1&0&0 \\ 1&0&1&0 \\ 0&1&1&0 \\ 1&1&1&0 \\ 1&0&0&1 \\ 0&1&0&1 \\ 1&1&0&1 \\ 0&0&1&1 \\ 1&0&0&0 \\ 0&1&0&0 \\ 0&0&1&0 \\ 0&0&0&1 \end{bmatrix} \tag{4}$$

This kind of parity check codes has been recently used to protect FIR filters from the effects of SEUs [21]. The most intuitive and effective protection consists of adding one Hamming encoder and one decoder to each register of the digital circuit (Fig. 3). For an 8-bit width datapath, a (12,8) code would be needed, what implies 4 additional registers per tap plus one encoder and one decoder. However, the use of Hamming codes to protect FIR filters from the effects of soft errors has some drawbacks. First, the decoder of each register is in the critical path to the output and therefore it decreases the maximum frequency of operation of the circuit; and second, the area consumed in the case of using one encoder and decoder into each tap of the delay line can be higher than TMR in some cases, as reported in [21].

3.3. Linear digital state variable system approach (LDSV)

This technique was first introduced in [14], and uses a mathematical model in order to represent generic circuits and detect and correct errors when they happen. It is called *linear* since it focuses on circuits whose state (at instant *n+1*) is a linear function of the previous states (at instant n) and the inputs.

With these considerations, such circuits may be represented in the following way:

$$S(n+1)^{\mathrm{T}} = A \cdot S(n)^{\mathrm{T}} + B \cdot U(n)^{\mathrm{T}} \tag{5}$$

where S is the state vector, U is the input vector and A, B are the coefficient matrixes that form the linear relationship. If this

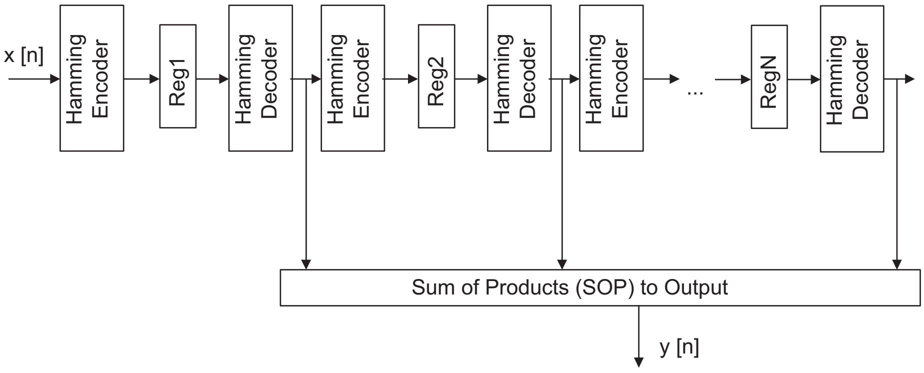

Fig. 3. FIR filter protection implementation using Hamming codes.

formula is developed then the following is obtained:

$$s_i(n+1) = \sum_{j=1}^{N} a_{ij}s_j(n) + \sum_{j=1}^{m} b_{ij}u_j(n) \tag{6}$$

where N is the number of states in the system, and m the number of inputs.

The idea of this approach is to add some checksum codes to these expressions in order to detect (and potentially correct) errors when they happen. The checksum codes used are called *real number codes*, which were introduced in [22]. The calculation of these codes will be explained next.

First, the X and Y matrixes are calculated as

$$X = CV \cdot A \quad \text{and} \quad Y = CV \cdot B, \tag{7}$$

where CV is the *coding vector*, an arbitrary vector used as a code generator. In this way, a checksum state variable can be defined as

$$c(n+1) = X \cdot [S(n)]^{\mathrm{T}} + Y \cdot [U(n)]^{\mathrm{T}}. \tag{8}$$

If this check state is added to Eq. (6) and developed, the following expression is obtained:

$$\begin{bmatrix} s_1(n+1) \\ s_2(n+1) \\ . \\ . \\ s_N(n+1) \\ c(n+1) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & . & . & a_{1N} & 0 \\ a_{21} & a_{22} & . & . & a_{2N} & 0 \\ . & . & . & . & . & 0 \\ . & . & . & . & . & 0 \\ a_{N1} & a_{N2} & . & . & a_{NN} & 0 \\ x_1 & x_2 & . & . & x_N & 0 \end{bmatrix} \cdot \begin{bmatrix} s_1(n) \\ s_2(n) \\ . \\ . \\ s_N(n) \\ c(n) \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} & . & . & b_{1m} \\ b_{21} & b_{22} & . & . & b_{2m} \\ . & . & . & . & . \\ . & . & . & . & . \\ b_{m1} & b_{m2} & . & . & b_{mm} \\ y_1 & y_2 & . & . & y_m \end{bmatrix} \cdot \begin{bmatrix} u_1(n) \\ u_2(n) \\ . \\ . \\ u_m(n) \end{bmatrix} \tag{9}$$

This represents the integrated model of the state, input and checksum code of the system.

Based on these expressions, several checks can be performed in order to identify and correct errors. A detailed explanation of them is out of the scope of this paper (see [14]).

For example, it can be proven that

$$c(n+1) = CV \cdot S(n+1)^{\mathrm{T}} \tag{10}$$

is always true, by construction of c.

Then, using Eqs. (8) and (10), errors can be detected as follows: at instant n, c is calculated and stored dynamically following equation (8) (this would correspond to the expected behavior); then, at instant n+1, it is re-computed using Eq. (10), and the obtained value (actual behavior) is compared with the previously stored one. If both are different, that would mean that an error has occurred. Data checksum and comparison circuitry are required in order to perform the mentioned operations.

The approach is then extended to also provide error correction capabilities by adding more checking vectors and additional logic [14].

4. Knowledge-based proposed technique

It is clear that any protection mechanism added to a circuit will incur a higher area in exchange for its extra functionality. Traditional techniques, as the ones explained in the previous section, usually try to achieve this protection level focusing strictly on the circuit itself. However, the same circuit, implemented in different applications and under different conditions, may require distinct protection levels. If, instead of always aiming at the same protection level, custom-tailored solutions are studied, taking the application and environment requirements into account, then the extra hardware added to achieve this protection will be minimal. This is what we call, in a generic way, to apply the *system knowledge*. Although this design philosophy can be extrapolated to any kind of circuit, the proposed technique (first introduced in our previous work [23] and continued in [24]) will be applied to general FIR filters in this paper.

The motivation to study this kind of filters is their high presence in Space application, due to the importance of signal processing in this environment [4,25,26]. Because of this intensive use and the criticality of operations, a reliable protection against the effect of radiation is fundamental.

These filters consist of one set of interconnected shift registers, together with some adders and multipliers which performs a specific operation to the input signal represented by the next equation.

$$y[n] = \sum_{i=0}^{N-1} x[n-i] \cdot h[i] \tag{11}$$

Two possible structures for FIR filter implementation are depicted in Fig. 4.

The knowledge-based proposed technique represents one alternative to the use of TMR in all registers taking advantage of the fact that the registers for the FIR implementation are connected in such a way that their values do not suffer changes as they move across the delay line. This can be used to compute a two-dimensional parity as follows:

- For each input value, compute a parity bit, named Pv (vertical). This bit is stored with the input value and it moves across the delay line. So, an extra register to store the Pv bit per tap in the delay line is needed.

- For each bit position in the input value, compute another parity bit, known as Ph (horizontal), across all the bits that have that position on the registers of the delay line. Ph values are stored in another set of registers.

Pv is only computed once, when the input arrives and enters the delay line. However, Ph is updated every clock cycle with the bit of the new value entering the delay line and the one leaving it. These two sets, Pv and Ph, form the accumulated parity of the circuit, which is constantly being updated (See Fig. 5).

For the example shown in Fig. 5, with an input word of four bits and four taps in the delay line, sixteen one-bit registers to store data bits would be needed, plus eight more single registers, four to store Pv parity bits and other four for Ph parity bits.

Dynamically, each time a new value reaches the circuit, both the horizontal and vertical parity are re-checked and compared with the accumulated values. Depending on the result of this comparison, an error may have been found in the system, which will potentially be eliminated using the correction logic in Fig. 5. However, a careful analysis has to be done, since misleading interpretations may occur. This analysis will be explained in the following paragraphs.

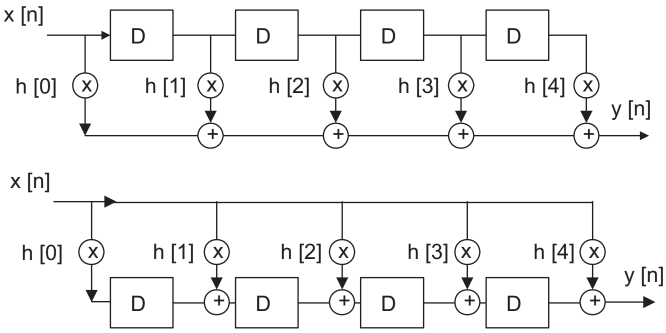

Fig. 4. Different structures for FIR filters.

Notice that the added Ph and Pv bits may also be affected by SEUs, and therefore, they should also be protected from them. Taking this into account, several situations can arise. The possible errors could be classified into single SEU events (only one bit-flip per cycle), and multiple SEUs (more than one bit-flip per cycle).

Considering the single SEU scenario, only two possibilities need to be considered:

- A single error has hit a data bit in the delay line. This should be the most usual case, since data bits are predominant in the system.
- A single error has hit one of the parity bits (Ph or Pv). This case can be misleading if not analyzed properly.

On the contrary, the multiple SEU situations (also called MBU) can adopt a countless number of possibilities, and a detailed analysis is out of the scope of this paper, as it will be commented below. In this case, any number of errors can happen at the same time, which may affect both data and parity bits.

Considering the different status of Ph and Pv, different error scenarios can happen:

1. No errors. The actual and accumulated values are the same. There is no problem with the system and its behavior can be taken as correct.
2. Single SEU. Three scenarios:
   (a) There is a discrepancy between a bit of the accumulated and actual Ph and between a bit of the accumulated and

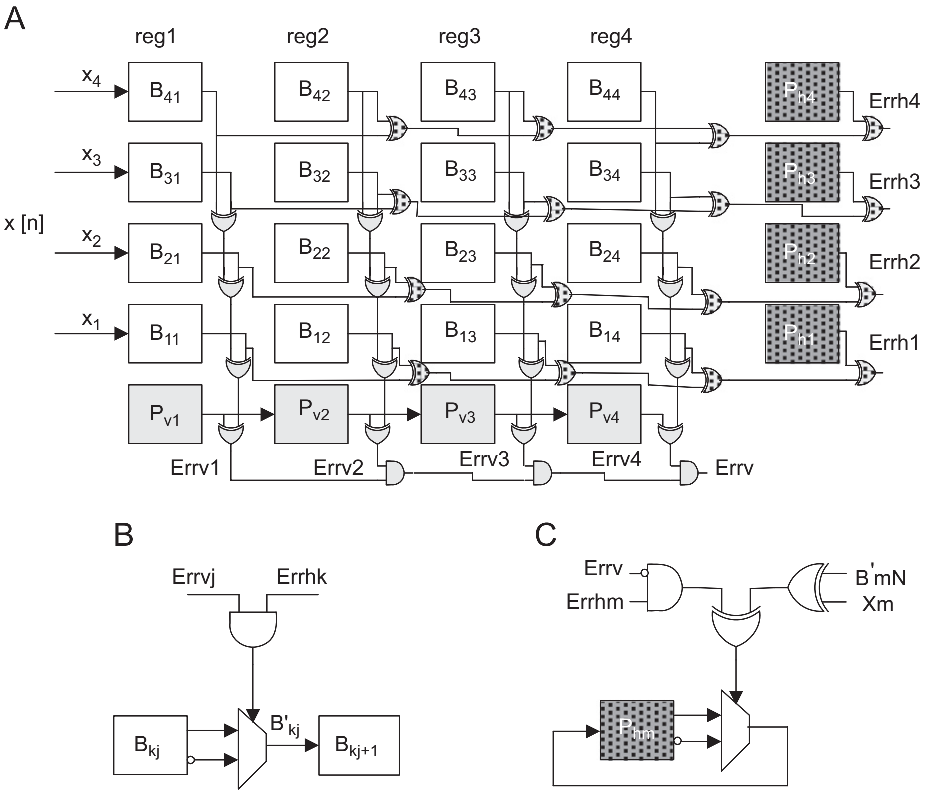

Fig. 5. Proposed implementation for protection against SEUs in FIR structures.

actual Pv. If both differences happen, that means a SEU has affected a register in the delay line, specifically, the bit located at the crossing point of the mentioned Ph and Pv (see correcting logic in Fig. 5B).

(b) There is a discrepancy between a bit of the stored and actual Ph value, but the accumulated and actual Pv bits are identical. If this happens, the SEU has occurred in the stored Ph that is discrepant, and should be corrected (see correcting logic in Fig. 5C).

(c) Same situation as the previous one but the discrepancy is in Pv (while Ph remains correct). Then, the SEU has affected that Pv. In this case, the Pv bit is not corrected, since this value will be shifted out of the delay line (together with the data bits), and the cost overhead incurred by the correcting logic can be saved. However, if while that Pv error is present, another SEU causes a different error, for example in a given Ph (producing a 2.b scenario), the combination of both events will confuse the system into a false 2.a scenario, triggering a wrong data correction. Nevertheless, considering the very low probability of such consecutive SEUs, this risk is acceptable if compared to the savings in correcting logic.

3. Multiple SEUs:
There is a high number of combinations produced by more than one SEU happening at the same time. A detailed analysis of these situations would be too long, and besides, it would not lead to a conclusive result, since SEUs are not univocally located in a multiple-error situation, just by examining the Ph and Pv bits. For example, if there is a discrepancy between two or more bits of the accumulated and actual Pv and between two or more bits of the accumulated and actual Ph, there is no way to determine exactly where the errors have happened. For a detailed list of scenarios affected by multiple bits, see [24].
Therefore, all double errors in data bits are never corrected, but this is not a major drawback after examining the problem closer, for a couple of reasons:
   - The probability of multiple (simultaneous) SEUs is reasonably low, as stated in several research sources, as [27]. This means that the number of uncorrected error will also be low.
   - No technique is 100% safe against multiple SEUs. For example TMR could suffer as much as the other techniques, as the redundant flip-flops would normally be close together, and therefore, if a double SEU happens, it is likely that two of the three TMR voters in a single bit are the affected ones.

Another different problem of this technique, which is also present when using Hamming codes, is the extra addition of combinational logic to the critical path that increases its delay, reducing the maximum operation frequency of the protected circuit. An interesting question is related to what happens if the error correction is active so late in the clock cycle, that correction cannot take place in time, and the error is propagated to the next stage of the delay line. This implies a time percentage of the clock cycle when the system is vulnerable to the error propagation. Next section includes explanations to these considerations.

5. Experimental results

In this section, the different protection techniques (TMR, Hamming codes and the proposed one) will be studied, in terms of area, protection effectiveness and critical path vulnerability. These techniques have been implemented in VHDL and then synthesized for a commercial ASIC library. Also, a study on the area cost of the LDSV technique, compared with the proposed one, will be provided.

Three experiments have been carried out on the circuits:

1. Using a simulation platform, several SEUs campaigns have been inserted, and the effectiveness of the protection techniques has been put in perspective. This simulation platform has been built in order to easily simulate SEUs in circuits. It uses Modelsim to hold the VHDL description of the circuit, Matlab to generate the reference input and output signals (which are compared with the actual behavior of the system to determine its correctness), and the Single Event Upset Simulation Tool (SST) developed at the European Space Agency [11] to insert SEUs and study the response of the circuit.
2. The circuits have been synthesized, and their complexity has been compared.
3. A study of the vulnerability that could cause error propagation in the delay line has been performed for the proposed technique and Hamming.

In this way, the quality of the proposed techniques is measured in terms of effectiveness and complexity.

The circuit chosen for the comparison process of the examined techniques to deal with the soft errors that generate bit flips in storage elements is the low pass FIR filter evaluated in [21]. The coefficient values for this specific filter in the Eq. (11) are:

$$h[n] = [-1 \quad 24 \quad 50 \quad 50 \quad 24 \quad -1] \tag{12}$$

The selected FIR filter has simple and symmetric coefficients, which reduces the complexity of multipliers (they are optimized to multiply constant values) and allows sharing them between taps of the delay line. Moreover, this structure is generic enough to consider the extracted results of the study as general conclusions about the compared techniques, considering general FIR implementations.

Fig. 6 shows the structure of the low pass FIR filter used in the experimental process.

In all the experiments, 8-bit input and output signals, $x[n]$ and $y[n]$, are considered. In the case of Hamming, the code used was (12,8).

5.1. Experimental environment description

As mentioned before, a simulation-based fault injection platform [12] has been used to evaluate the effectiveness of the proposed technique (see Fig. 7).

The platform is composed of the SST simulator developed by the ESA Data Systems Division and Matlab. A commercial HDL simulator (ModelSim) is used to run the simulations. For a given circuit under test, a number of test cases in terms of the corresponding input and output data would be generated using Matlab. Also, a number of test configurations in terms of the soft errors inserted using the SST would be produced. The test cases

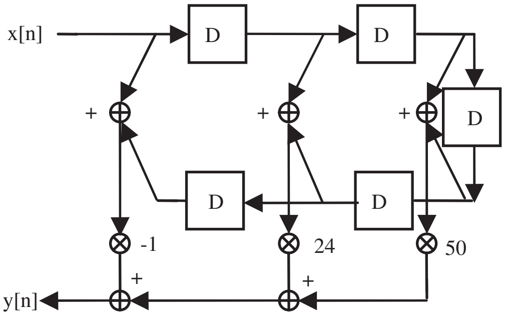

Fig. 6. Low pass FIR filter structure.

would be designed to fully test the circuit functionality and performance while the test configurations would ideally reflect the soft error environment that is expected for the device operating conditions. Then, combinations of both can be easily tested by just selecting the appropriate input and output data files and test configuration file. In fact, with a simple script, the testing of all relevant combinations can be easily automated.

The main modules of the platform can be seen in Fig. 7 and are briefly described below:

1. *SEUs Simulation Tool (SST)*: This component consists of a set of modules used to prepare the environment to generate soft errors in both sequential and combinational logic.
2. *HDL Simulator*: This module is in charge of holding the circuit to test, and performing a simulation at the design stage. The description of the environment is divided into two parts: the circuit and the test bench. In particular, the test bench will produce the different test scenarios for the circuit (based on the input values provided by the Matlab module), will capture the circuit outputs, and will compare them with the expected results (also provided by Matlab). In the case both are different, that will indicate an error, which will be logged in the system for further study. This environment is generic (independent of the circuit behavior) for circuits devoted to signal processing (or at least a significant part of them). It is also flexible in the way that it is straightforward to generate different input signals to test the circuit operating in several environments. For other kinds of circuits (e.g., controllers), another application rather than Matlab would be designated to hold the golden data.
3. *Matlab*: This module compares the theoretically correct behavior of the system with the actual outputs produced by the HDL simulator. It has the advantage that the Matlab code does not need to reflect the actual circuit implementation, it only needs to be functionally equivalent. This facilitates the use of a single Matlab model to explore different implementation alternatives. The difference between both behaviors will indicate the presence of a SEU, what will trigger the mechanism to detect the source of such an error.

5.2. *Effectiveness*

Using the platform described above, a campaign of tests has been performed in order to evaluate the behavior of the system when protected with the different techniques. The procedure is as follows: first, a random input sequence of 15,000 samples formed by pulses plus noise has been generated through Matlab, and then several sequences of 100 SEUs in random time instants have been inserted into the different techniques. These instants were selected using an equally distributed probability function, with the particularity that the maximum number of bit flips that can occur during each clock cycle is one.

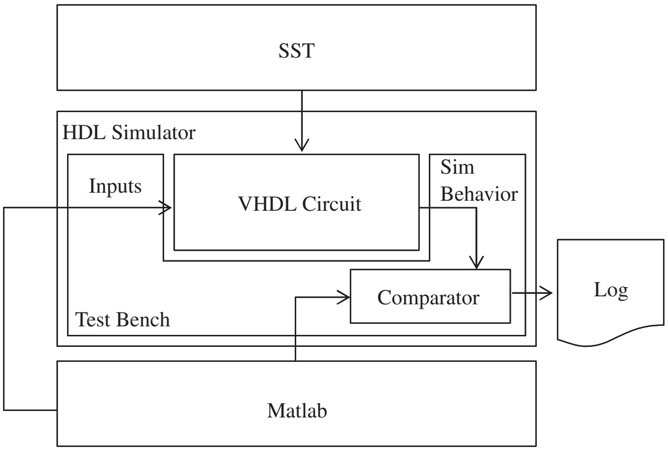

Fig. 7. Scheme of the simulation-based platform.

Results of these tests on the three protection techniques show the same conclusions: the three techniques are totally effective against single SEUs (no errors propagated to the output).

5.3. *Complexity*

In order to compare the complexity of the knowledge-based proposed technique with TMR and Hamming codes, the three systems have been synthesized. The area results in equivalent gates (see Table 1) have been generated for a TSMC 0.25 um library using Leonardo (by Mentor), assuming a 50 MHz clock and an 8-bit datapath.

As it can be seen in the table below, the proposed technique is the one with less area cost, followed by the Hamming protection methodology.

These experiments have been repeated for a larger filter, with 10 taps. The increment of area for the three techniques is illustrated in Table 2. In this case, the coefficients for the FIR filter with ten taps are shown in Eq. (13):

$$h[n] = [-1 \quad 3 \quad 50 \quad 64 \quad 96 \quad 96 \quad 64 \quad 50 \quad 3 \quad -1] \qquad (13)$$

Again, the proposed technique has the lowest area cost, what implies that it is scalable as the size of the filter grows.

5.4. *Vulnerability*

Another interesting issue related to the correction logic of the proposed technique and Hamming consists of the time that the logic needs to execute the correction. Due to the extra combinational logic added, if the correction comes so late in the cycle, the error will not be corrected, and it will go through the delay line.

Let us define the vulnerability of the technique as the percentage of the cycle when a sudden error is not corrected, due to the time issues discussed before. In other words, the vulnerability is associated to the delay introduced by the extra hardware of the protection logic, what can imply that the correcting signals do not arrive in time (if these are triggered late enough in the cycle), and therefore producing the error propagation. In this way, if e.g. the vulnerability is 25%, that means that a SEU would be corrected during the first 75% of the cycle time, but it will be propagated to the next stage if it happens in the last 25%. Obviously, the lower the vulnerability, the better it is. Table 3 shows the results after comparing the vulnerability of the proposed technique and Hamming, by performing an analysis on the critical paths. Two experiments have been conducted, with frequencies of 10 and 100 MHz. As it can be seen in this table, the

Table 1
Comparison of TMR and Hamming with the proposed technique for a six-coefficient filter

FIR with 6 taps	Frequency	Gates	Gate overhead on unprotected (%)
(a) FIR with TMR	117.1	1704	115.4
(b) FIR with Hamming	105.5	1476	86.6
(c) FIR using system knowledge	104	1363	72
FIR without redundancy (unprotected)	137.8	791	–

Table 2
Comparison of TMR and Hamming with the proposed technique for a ten-coefficient filter

FIR WITH 10 TAPS	Frequency	Gates	Gate overhead on unprotected (%)
(a) FIR with TMR	104.2	2619	104.8
(b) FIR with Hamming	94.6	2405	88
(c) FIR using system knowledge	90.4	2114	65.3
FIR without redundancy (unprotected)	114.4	1279	–

Table 3
Vulnerability of Hamming and the proposed technique due to extra logic in the critical path

Frequency (MHz)	FIR with Hamming (%)	FIR using system knowledge (%)
100	21	14
10	2.1	1.4

Table 4
Final comparison

	Total area	Effectiveness	Delay
TMR	High	Good	N/A
Hamming	Average	Good	Average
Proposed technique	Best	Good	Good

vulnerability is worse for Hamming codes than for the proposed technique. In both cases, the relative reduction of the vulnerability has been 33%, what means that the proposed technique will be able to correct SEUs arriving at a later instant than Hamming. Although this reduction may seem small, it has an importance influence in the behavior of the circuit, since the error rate propagated at the output would statistically be reduced in the same percentage.

Finally, to conclude this section, a summary of all the conducted experiments in this paper is offered in Table 4. This allows a quick comparison of the three techniques for area, effectiveness and vulnerability.

5.5. Area cost: proposed technique vs. the LDSV approach

As a final experiment, the proposed technique needs to be compared with the methodology presented in Section 3.3, the linear digital state variable system approach. The problem with this approach is that although good results have been reported using it, nothing is said about its implementation cost, in terms of the coefficient matrixes and the rest of circuits needed to perform the calculations.

Let us analyze how this technique would be applied to the case study of this paper. We need to consider that for the FIR expression in Eq. (11), the relationship between the state and the system input is

$$\begin{bmatrix} s_1(n) \\ s_2(n) \\ . \\ . \\ s_{N-1}(n) \end{bmatrix} = \begin{bmatrix} x(n-1) \\ x(n-2) \\ . \\ . \\ x(n-N+1) \end{bmatrix} \tag{14}$$

This is because the main structure of the filter consists of a delay line that simply shifts the input through the different taps.

Using Eqs. (11) and (14), and in order to satisfy Eq. (9), the A and B matrixes may be derived as

$$A = \begin{bmatrix} 0 & 0 & . & 0 & 0 \\ 1 & 0 & . & 0 & 0 \\ 0 & 1 & . & 0 & 0 \\ . & . & . & . & . \\ 0 & 0 & . & 1 & 0 \\ 1 & 1 & . & 1 & 0 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 1 \\ 0 \\ 0 \\ . \\ 0 \\ 1 \end{bmatrix} \tag{15}$$

where the last rows of A and B correspond to the check codes of X and Y, respectively.

A coding vector $CV = (1, 1,\ldots,1)$ has been used. Note that in this case, the input vector U is just the scalar $x(n)$.

The first conclusion of this study is that both the A and B matrixes are sparse (most of their values are null) when calculated for the FIR filter. This is due to the particular FIR structure, whose delay line forces the regular distribution of matrix A: the diagonal of 1's represents the information shift through the filter.

It is clear that when A and B are sparse, the overhead of computing *c(n)* using Eq. (8) and checking it with Eq. (10) can be significant. For the FIR filter under study, even when using the simple coding vector $CV = (1, 1, \ldots, 1)$, $N-1$ adders (where N is the number of filter taps) are needed to compute each equation, plus the registers needed to store *c(n)* and the logic to compare the stored value with the re-computed one using Eq. (10). For this particular case, this adds up to 10 adders, 11 registers and one 11-bit comparator. The LDSV approach has been applied as per Eq. (15) to the filter under study and synthesized, providing a gate count of 1457 (larger than the one associated to our proposed technique) and just to provide only error detection. For error correction, more checking vectors and additional logic would be needed. Moreover, the checking vectors have to include different values to be able to provide error correction so that multipliers will be used instead of adders. All these results clearly show that a higher area overhead is required, compared with the proposed technique, Hamming or even TMR (see Table 1).

In addition to the previous discussion, the vulnerability introduced in Sections 4 and 5.4 will also be higher for this approach. The error detection logic delay is given by the delay of computing Eq. (10) plus the comparison operation, which is clearly larger. For error correction, more logic is needed and therefore the delay is even larger. Due to its complexity, a detailed analysis of the vulnerability for this technique is outside of the scope of this paper, but it can be seen that as for the gate count, the vulnerability will be much higher than for Hamming or the proposed technique.

Therefore, this reinforces the idea of why ad-hoc techniques like the one proposed in this paper would lead to more optimal results than generic approaches, as the one described in [14].

6. Conclusions

In this paper, a new protection technique for FIR filters has been presented. Several experiments have been conducted in order to compare this technique with TMR and Hamming codes. It has been showed that the new technique, while providing a similar protection level to the others, incurs a lower area overhead, which makes it more convenient.

Also, the presented technique has been compared with a generic and well-known approach (LDSV) based on check code correction and detection, in terms of area complexity. It has been proven that although these techniques are easily implemented on general circuits, they do not lead to an optimal cost as ad-hoc techniques would do.

Other conclusions that can be drawn from the presented work are:

- The proposed knowledge-based technique can be easily extended to work with other kinds of digital structures. In other words, this approach is not generic for a particular type of circuits, but its generalization can be done easily.
- From the obtained results, adaptive filters would be good candidates to continue exploring this technique. The reason of this is since these filters have a larger area, this would produce even more savings than the traditional protection techniques.
- Finally, the presented technique is not restricted to Space applications. Other statistical functions to foresee soft errors rates in different environments could be utilized, what would enable to model other kind of scenarios.

References

[1] J.E. Mazur, An overview of the space radiation environment, The Aerospace Corporation Magazine of Advances in Aerospace Technology, vol.4, No. 2, Summer, 2003.

[2] D. Schrimpf, D.M. Fleetwood, Radiation Effects and Soft Errors in Integrated Circuits and Electronic Devices, World Scientific Publishing, 2004.

[3] P.E. Dodd, L.L. Massengill, Basic mechanisms and modeling of single-event upset in digital microelectronics, IEEE Trans. Nucl. Sci. 50 (3) (2003) 583–602.

[4] E.R. Prado, J.P. Prewitt, A high performance COTS based vector processor for space, in: Proceedings of the IEEE Aerospace Conference 2000, pp. 227–233.

[5] S. Mitra, N. Seifert, M. Zhang, Q. Shi, K.S. Kim, Robust system design with built-in soft-error resilience, IEEE Comput. 38 (2) (2005) 43–52.

[6] U. Krautz, M. Pflanz, C. Jacobi, H.W. Tast, K. Webe, H.T. Vierhaus, Evaluating coverage of error detection logic for soft errors using formal methods, in: Proceedings of the Design Automation and Test Conference 2006 (DATE’06), pp. 176–181.

[7] M.P. Baze, S.P. Buchner, D. McMorrow, A digital cmos design technique for seu hardening, IEEE Trans. Nucl. Sci. 47 (6) (2000) 2603–2608.

[8] S. Hanbic, FTMR: Functional Triple Modular Redundancy, ESA Report FPGA-003-001, December 2002.

[9] B. Shim, N.R. Shanbhag, Energy-efficient soft error-tolerant digital signal processing, IEEE Trans. Very Large Scale Integrat. (VLSI) Syst. 14 (4) (2006) 336–348.

[10] A. Reddy, P. Banarjee, Algorithm-based fault detection for signal processing applications, IEEE Trans. Comput. 39 (10) (1990) 1304–1308.

[11] D. Gonzalez-Gutierrez, Single even upset simulation tool functional description, ESA Report TEC-EDM/DCC-SST2, July 2004.

[12] O. Ruano, J.A. Maestro, P. Reyes, P. Reviriego, A simulation platform for the study of soft errors on signal processing circuits through software fault injection, in: Proceedings of the IEEE International Symposium on Industrial Electronics 2007 (ISIE’07), pp. 3316–3321.

[13] J.F. Ziegler, Terrestrial cosmic rays, IBM J. Res. Develop. 40 (1) (1996) 19–40.

[14] A. Chatterjee, M.A. d’Abreu, The design of fault-tolerant linear digital state variable systems: theory and techniques, IEEE Trans. Comput. 42 (7) (1993) 794–808.

[15] R.R. Rao, K. Chopra, D. Blaauw, D. Sylvester, An efficient static algorithm for computing the soft error rates of combinational circuits, in: Proceedings of the Design Automation and Test Conference 2006, (DATE’06), pp. 164–169.

[16] B.S. Gill, C. Papachristou, F.G. Wolff, Soft delay error analysis in logic circuits, in: Proceedings of the Design Automation and Test Conference 2006 (DATE’06), pp. 47–52.

[17] S. Almukhaizim, Y. Makris, Concurrent error detection in asynchronous burst-mode controllers, in: Proceedings of the Design Automation and Test Conference 2005 (DATE’05), pp. 1272–1277.

[18] J. Kumar, M.B. Tahoori, Use of pass transistor logic to minimize the impact of soft errors in combinational circuits, Workshop on System Effects of Logic Soft Errors 2005 (SELSE’05).

[19] S. Mitra, T. Karnik, N. Seifert, M. Zhang, Logic soft errors in sub-65 nm technologies design and CAD challenges, in: Proceedings of the Design Automation Conference 2005 (DAC’05), pp. 2–4.

[20] P. Sweeny, Error Control Coding, From Theory to Practice, Wiley, 2002.

[21] R. Hentschke, F. Marques, F. Lima, L. Carro, A. Susin, R. Reis, Analyzing area and performance penalty of protecting different digital modules with Hamming code and triple modular redundancy, in: Proceedings of the 15th Symposium on Integrated Circuits and Systems Design, 2002, pp. 95–100.

[22] V.S.S. Fair, J.A. Abraham, Real-number codes for fault-tolerant matrix operations on processor arrays, IEEE Trans. Comput. 39 (4) (1990) 426–435.

[23] P. Reyes, P. Reviriego, J.A. Maestro, O. Ruano, New protection techniques against SEUs for moving average filters in a radiation environment, IEEE Trans. Nucl. Sci. 54 (4) (2007) 957–964.

[24] P. Reyes, P. Reviriego, J.A. Maestro, O. Ruano, A new protection technique for finite impulse response (FIR) filters in the presence of soft errors, in: Proceedings of the IEEE International Symposium on Industrial Electronics 2007 (ISIE’07), pp. 3328–3333.

[25] C. Lambert-Nebout, G. Moury, On-board digital signal processing for IASI mission, in: Sixth International Workshop on Digital Signal Processing Techniques for Space Applications, Noordwijk, The Netherlands, September 1998.

[26] S.M. Parkes, DSP (Demanding Space-based Processing!): the path behind and the road ahead, in: Sixth International Workshop on Digital Signal Processing Techniques for Space Applications, Noordwijk, The Netherlands, September 1998.

[27] H.C. Koons et al., The impact of the space environment on space systems, Space and Missile Systems Center-Air Force Materiel Command, Report No. TR-99(1670)-1, July 1999.

Juan Antonio Maestro holds a Ph.D. degree in Computer Architecture (1999) from Universidad Complutense de Madrid. He has served both as a lecturer and researcher at several universities, as Universidad Complutense de Madrid, UNED (Open University), Saint Louis University and Universidad Antonio de Nebrija, where he currently manages the Computer Architecture and Technology Group. Besides from this activity, he has worked for several multinational companies, managing IT projects and organizing support departments. His areas of interests range from High Level Synthesis and co-Synthesis to Signal Processing and Real-Time systems.

Pedro Reviriego holds a Ph.D. degree (1997) and a M.Sc. in Telecommunication Engineering (1994), both from Universidad Politécnica de Madrid. His professional experience has been developed in different companies, as Agere Systems (Distinguished Member of Technical Staff), Massana Technologies (Principal Engineer) and Teldat S.A. (Research Engineer), as well as a professor at Universidad Carlos III de Madrid. His areas of expertize include Signal Processing for Communications and Audio, Computer Networks, VLSI and Real-Time Systems. Among the projects he has worked for, the DIGISAT (Digital Services for Communication Satellites) ACTS project and the Small Satellite programme (UPM) should be highlighted.

Pilar Reyes holds a M.Sc. in Telecommunication Engineering (2004) from Universidad de Sevilla, as well as a degree in Industrial Engineering (2000) from Universidad de Córdoba. She has been a researcher with Anafocus, working on the design an implementation of a video processing system. Currently, she is a full-time researcher at Universidad Antonio de Nebrija, as well as a Ph.D. candidate at the Universidad Complutense de Madrid doctoral programme.

Oscar Ruano holds a M.Sc. in Computer Engineering (2005) from Universidad Antonio de Nebrija. He has worked with different multinational companies in the IT consultancy field, as Accenture. Currently, he is a full-time researcher at Universidad Antonio de Nebrija, as well as a Ph.D. candidate at the Universidad Complutense de Madrid doctoral programme.