Efficient Structures for the Implementation of Moving Average Filters in the Presence of SEUs using System Knowledge

P. Reyes, P. Reviriego, *Member, IEEE,* O. Ruano and J.A. Maestro

***Abstract*— In this paper, new techniques for the implementation of *moving average filters* with protection against Single Event Effects (SEEs) are presented, incurring lower circuit complexity and cost than traditional techniques like Triple Modular Redundancy (TMR). The baseline of the presented approach is to exploit the *system knowledge* (i.e. the structure of the filter) to deal with SEEs at a higher level of abstraction, rather than the usual non-specific techniques.**

***Index Terms*— Single Event Upsets (SEUs), radiation hardening, digital filters, redundancy.**

I. Introduction

The effects of radiation on microelectronic circuits have a number of consequences that impact the design of devices operating in the presence of radiation [1]. One type of effects is Single Event Effects (SEEs) that cause changes in the values of flips-flops (SEUs) or combinational logic (SETs) [2]. To mitigate the effects of SEEs, a number of techniques can be used at the physical level (device size and structure) [3]. In addition to those techniques, redundancy can be introduced in the design so that it can detect and correct SEEs [4]. 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 [4]. One advantage of both TMR and FTMR is that they are general techniques that can be applied to most digital circuits. However, this comes at a high cost in terms of circuit area and power and more so for FTMR.

This work was supported by the Spanish Ministry of Science and Education under Grant ESP-2006-04163 (partly financed with FEDER funds).

P. Reyes, O. Ruano and J.A. Maestro are with Universidad Antonio de Nebrija C/ Pirineos, 55 E-28040 Madrid, Spain (phone: +34 914521100; fax: +34 914521110; email: {jmaestro,preyes,oruano}@nebrija.es).

P. Reviriego is with Universidad Carlos III de Madrid. Av. Universidad, 30 E-28911 Leganés, Spain (email: revirieg@it.uc3m.es).

On the contrary, the approach to deal with SEEs in this paper is to apply *circuit specific techniques* that exploit the inherent redundancy or fault tolerance of some circuits [5],[6], 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 cannot achieve.

In order to prove this approach, Digital Finite Impulse Response (FIR) filters have been chosen due to the high presence of these structures in the communication systems of most application areas. Particularly, a specific type of FIR filters has been used, the *moving average filter*, which performs the following operation [7]:

$$y[n] = \frac{1}{N}\sum_{i=0}^{N-1} x[n-i] \quad (1)$$

A number of structures have been proposed to implement this kind of FIR filters [7]. Basically, the most common ones are: i) implement the FIR filter in a direct way, ii) implement it through an IIR structure (see Fig. 1).

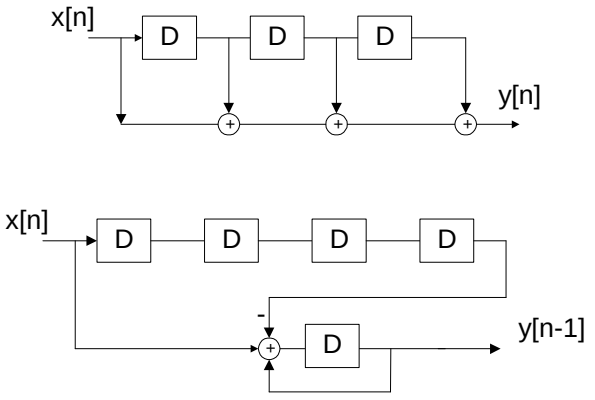

Fig. 1: A) FIR Structure (top) and B) IIR Structure (bottom).

The main difference is that while the FIR structure is more resilient to SEUs, with temporary effects, the IIR structure is much simpler (and cheaper), but more vulnerable to SEUs that have permanent effects.

The research presented in this paper is oriented to get the IIR structure knowledge in order to propose an efficient implementation that:

1) Has a suitable protection level against SEUs for the given circuit.
2) Has a cost (complexity) lower than both the IIR protected with TMR and the FIR structure.

II. PROPOSED TECHNIQUES

To come up with optimal SEEs protection techniques, the requirements of the application in which the filter is used need to be taken into account. In this section, three application scenarios are described, with a protection technique specific for each of them.

A. Technique 1

Filter to detect Ethernet Link Pulses [8] in the presence of noise, such that idle periods happen in between pulses. Due to the control logic of this application, errors can be tolerated at the output of the filter as long as they are *transient*. In this application, both the FIR-like implementation and the IIR-like implementation could be used. However, some protection techniques should be added to the latter, since the effects of SEUs on it are *permanent*. The traditional approach to deal with SEUs would be to apply TMR to all storage elements for a total protection. However, it would provide a protection level higher than needed (transient errors are allowed). Due to this, and in order to reduce the inherent complexity of TMR, a specific approach should be used applying the system behavior, what would lead to a less costly solution.

In this way, an extra counter can be added, which will compute the number of consecutive idle cycles in the system. By idle, we mean cycles in which the system input is below a given threshold, what would indicate the presence of noise in the filter. On the other hand, inputs over the noise threshold would be considered as active data. Therefore, if the counter detects N+1 consecutive idle cycles (being N the number of taps in the circuit), this would mean that anything inside the circuit (stored in the delay line) comes from acquired noise, and therefore can be discarded (reset). With this sanity check mechanism, the filter actually is reset whenever N+1 idle cycles are detected, what has an indirect benefit as protection technique for SEUs. No matter when a SEU hits the system, it will only last until the next reset of the circuit, making its effect transient. Notice that the mechanism will only work if frequent N+1 idle cycles can be guaranteed, but since the system knowledge assure that this application can expect this kind of input, the proposed technique will be good enough, with a cost much lower than TMR.

B. Technique 2

Application that can tolerate occasional errors at the output of the filter (like in scenario 1) but there is no guarantee of idle periods that can be used to reset the system. Therefore, the previous technique is no longer valid.

In this situation, the computation of the output can be done in parallel by another structure added to the filter, for the sake of comparison. Obviously, if this added structure is a replica of the filter itself, we would be doubling the complexity of the system. To avoid this, this parallel structure will be implemented with a *decimated* filter, which has a structure simpler than a regular filter, with the drawback that it only computes the right output one out of N cycles.

In this way, the output of both structures would be compared at each y[n*N], what is one out of N times. After the comparison, several cases can arise:

- Both structures have the same output. This means the system is error-free.
- The structures have different outputs. This means that a SEU has hit the original filter or the secondary (decimated) one. To determine where the error is, the decimated filter is reset, and after N cycles the comparison is made again. After this, it may happen that:
    - The error is gone. That means the SEU was in the decimated filter.
    - The error is still present. That means the SEU is in the main filter and needs to be reset, in order to eliminate it.

Regardless of which of the previous situations happens, the SEU will always be transient, what satisfies the application requirement, but in this case, no idle periods are needed. Therefore, the protection level has been increased with a minimum complexity increment thanks to applying the knowledge of the system.

C. Technique 3

Application demanding such a protection level that SEUs do not cause any errors at the output of the filter. This protection level is similar to the one provided by TMR. Therefore, one alternative would be to use TMR itself in all registers. However, instead of a massive triplication, a less complex approach would be to compute a two-dimensional parity structure as follows. For each input value a parity bit Pv (vertical) is computed, and for each bit position of the input, another parity bit Ph (horizontal) across all the registers of the delay line is computed. Pv is only computed 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.

Dynamically, each time a new value reaches the circuit, both the horizontal and vertical parity are re-checked and compared with the accumulated values. Then, several situations can arise:

- The actual and accumulated values are the same. There is no problem with the system and its behavior can be taken as correct.
- There is a discrepancy between a bit of the accumulated and actual Ph *and* between a bit of the accumulated and actual Pv. If both differences happen, that means a SEU has affected a register in the delay line. The bit affected by the SEU is the crossing point of the discrepant Ph and Pv.

In this way, since it has been identified, it can be corrected instantly, and therefore, the system behavior remains correct.

- There is a discrepancy between a bit of the accumulated and actual Ph *or* between a bit of the accumulated and actual Pv. If only one of the parity registers shows the discrepancy, it would mean a SEU has affected the discrepant parity register itself, and therefore it should be corrected. It is important to remember that all the extra structure added for protection can also be affected by SEUs.

The conclusion is, reviewing all the possible cases, that this technique detects and correct any SEU in the system, giving a protection level similar to TMR.

III. Experimental Results

In this section, the quality of the presented techniques will be studied. These techniques have been implemented in VHDL and then synthesized for a commercial ASIC library. Two experiments have been carried out on the circuits:

1) Using a simulation platform[1], several SEUs campaigns have been inserted, and the effectiveness of the protection techniques has been put in perspective. More than 20 test scenarios were reenacted, what implies over 200 impacts of SEUs.
2) The circuits have been synthesized, and their complexity has been compared with the traditional protection techniques.

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

A. *Effectiveness*

From the whole set of test scenarios, a particular case has been chosen for each technique in order to depict the obtained behavior of the system. These results can be extrapolated to the rest of the cases.

To test the first example of protection techniques described in the previous section, an 8-bit input signal, x[n], with range [-1,1] and N=16 are considered. The threshold to reset the accumulator is three LSBs. An initial sequence (see Fig. 2 Top) of pulses (instants multiple of 100) plus impulsive noise (instants 50, 150, 250, etc.) are generated through Matlab, and then, a SEU is introduced in the accumulator on cycle 20. The results (Fig. 3 bottom) show that there is no permanent error and also demonstrates how the circuit recovers from the wrong behavior when the protection techniques is used. On the other hand, the circuit without protection suffers a clear misbehavior, being the output sequence shifted by the effect of the SEU (Fig. 3 top).

[1] 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 [9] to insert SEUs and study the response of the circuit. A detailed description of this platform can be found in [10].

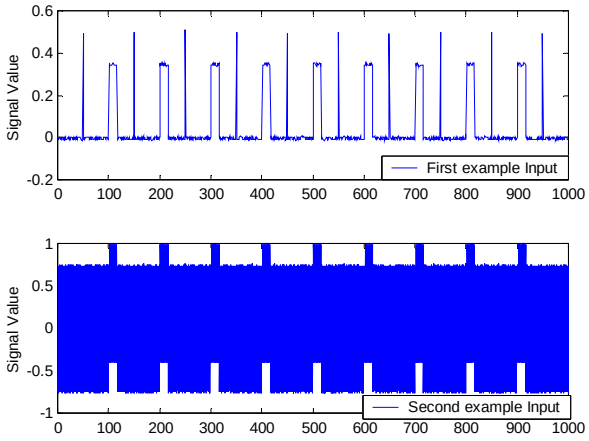

Fig. 2. Input signals for examples 1 (top) and 2 (bottom).

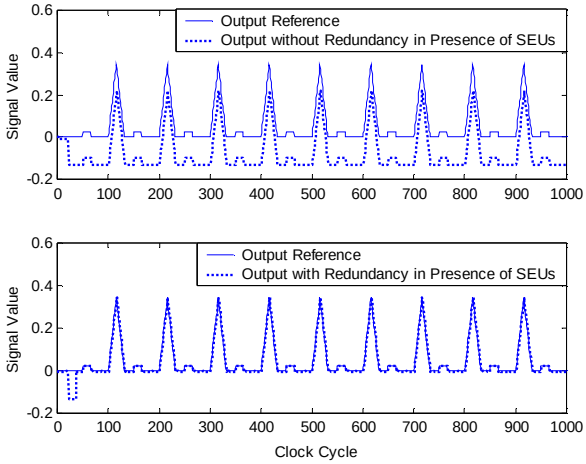

Fig. 3. Output of an unprotected IIR (top) and of an IIR protected with technique 1 (bottom).

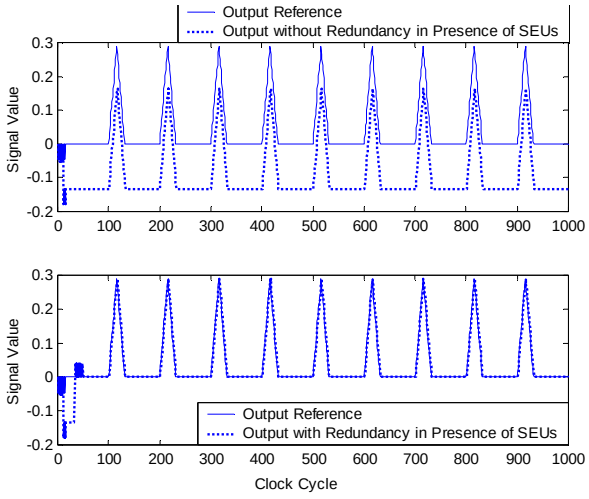

Fig. 4. Output of an unprotected IIR (top) and of an IIR protected with technique 2 (bottom).

For the second example, a signal that contains pulses plus a high frequency noise (Fig. 2 bottom) is generated. Then, the SST tool is used to insert a SEU in cycle 10. In this case, technique 1 would not work, as the incoming signal continuously exceeds the threshold (in other words, there is not any idle period). However, as it can be seen in Fig. 4 bottom, technique 2 removes the error in approximately 2*N samples. The circuit without protection shows a wrong behavior again (Fig. 4 top).

For the third example, some SEUs in the registers of the delay line have been introduced in order to check that they are corrected by the parity logic. Besides, some SEUs have been also introduced in the parity bits to check that erroneous corrections are not triggered. In all cases, the output of the filter does not show any discrepancy with the reference output. No plot of the results is provided since both the expected and actual outputs coincide.

B. *Complexity*

In order to compare the complexity of the proposed techniques with TMR, the three systems have been synthesized. The area results in equivalent gates (see Table I and Table II) have been generated for a TSMC 0.25um library assuming a 50 Mhz clock and an 8-bit datapath.

TABLE I
NUMBER OF GATES FOR THE FIR LIKE IMPLEMENTATIONS

	N = 8	N = 16	N = 32
FIR	713	1591	3788
FIR with TMR	1686	3538	7500

TABLE II
NUMBER OF GATES FOR THE IIR LIKE IMPLEMENTATIONS

	N = 8	N = 16	N = 32
IIR	517	866	1550
IIR with Tech. 1	690	1141	2024
IIR with Tech. 2	835	1300	2186
IIR with TMR	1633	2962	5591
IIR with Tech. 3	1452	2332	4029

The first conclusion is that the IIR implementation is less complex than the FIR one, as expected, and that for large values of N, the difference can be more than double. This explains why the IIR implementation is normally used in the absence of SEUs. Besides, although not shown in the tables, the IIR implementation is also faster and therefore, for the same technology, the filter can operate at higher clock frequencies.

In the presence of SEUs, the unprotected FIR versus the IIR protected with techniques 1 and 2 are compared. It can be seen that for large values of N the IIR with techniques 1 and 2 is still significantly smaller than the FIR one. For N=16 the reduction is 28% and 18% respectively, while for N=32 the reduction is over 40% in both cases. This is so because for these techniques the amount of redundancy introduced is almost independent of the value of N.

Finally, in the presence of SEUs, the IIR protected with technique 3 (total protection) versus the IIR protected with TMR are compared. A direct conclusion is that technique 3 results in a reduction of 20% to 30% in the number of gates. These results show that technique 3 provides a similar level of protection than TMR, but with less area cost.

IV. CONCLUSIONS AND FUTURE WORK

In this paper, new techniques to protect moving average filter implementations from the effects of SEUs have been presented. These new techniques have been developed using both application and system knowledge to provide a more intelligent protection, rather than the traditional and general approaches. The benefits of applying the system knowledge are clearly proved with the experimental results that have been obtained: all the techniques incur a lower circuit complexity than the traditional TMR approach, with a similar protection level.

The use of a simulation environment that enables a flexible testing of the effects of SEUs on signal processing circuits will facilitate the next research steps, which will focus on more general FIR filters and the consideration of FPGA-based implementations.

ACKNOWLEDGMENT

The authors would like to thank D. Gonzalez-Gutierrez for his help with the ESA SST tool and for his comments and suggestions, and E. Nogales for her contribution to some of the experiments that are reported in this paper.

REFERENCES

[1] R. D. Schrimpf and D. M. Fleetwood, “Radiation effects and soft errors in integrated circuits and electronic devices”, World Scientific Publishing, 2004. ISBN: 981-238-940-7.

[2] P. E. Dodd and LL. Massengill, “Basic Mechanisms and Modeling of Single-Event Upset in Digital Microelectronics”, IEEE Transactions on Nuclear Science, Vol 50, No 3, June 2003.

[3] M. P. Baze, S. P. Buchner and D. McMorrow, “A Digital CMOS Technique for SEU Hardening”, IEEE Transactions on Nuclear Science Vol 47, No 6, December 2000.

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

[5] B. Shim, N. R. Shanbhag, and S. Lee, “Energy-efficient soft error-tolerant digital signal processing”, Signals, Systems and Computers, 2003. Conference Record of the Thirty-Seventh Asilomar, Nov. 2003.

[6] H. Yuang-Ming, E. E. Jr. Swartzlander, “FFT arrays with built-in error correction”, Signals, Systems and Computers, 1994. 1994 Conference Record of the Twenty-Eighth Asilomar Nov. 1994.

[7] A. V. Oppenheim and R. W. Schafer, “Discrete Time Signal Processing”, Prentice Hall 1999.

[8] IEEE 802.3i Ethernet over Unshielded Twisted Pairs Standard (10BaseT) 1990.

[9] D. Gonzalez, “Single Event Upset Simulation Tool Functional Description”, ESA Report TEC-EDM/DCC-SST2, July 2004.

[10] Computer Architecture and Technology Group, “Research Project on Circuit Design for Radiation Environments”, TR-2005Dec-002, Universidad Antonio de Nebrija, http://www.nebrija.es/~jmaestro/Research_Project_UAN.pdf