Fault Tolerant FIR Filters Using Hamming Codes

Shih-Fu Liu, Pedro Reviriego and Juan Antonio Maestro

Abstract— **Hamming Codes have been used to protect different circuits against Single Event Upsets (SEUs). In this paper, the use of Hamming on FIR Filters is studied in order to provide optimized and efficient protection techniques.**

***Index Terms*— Single Event Upset (SEU), Hamming codes, Reliability, Digital filters.**

I. Introduction

Since the early stages of microelectronics, radiation has been identified as a source of alteration of functionality of integrated circuits. As the technology is steadily evolving and shrinking from generation to generation, the effects of a charged particle striking the silicon surface of an integrated circuit are getting more and more significant [1]. These effects can be divided into two kinds: Single Event Effects (SEE) – temporary errors such as Single Event Upset (SEU) and Single Event Transient (SET) – and Total Ionizing Dose (TID), which leads to permanent damage of the transistor, or circuit functionality, such as Single Event Latchup (SEL), Single Event Gate Rupture (SEGR), or Single Event Burnout (SEB) [2][3].

When a particle hits the silicon, it loses its energy and transmits it to the silicon, causing a current burst which in the case of SEUs changes randomly the content of storage cells. To protect storage cells of integrated circuits from this phenomenon, several approaches may be followed. One is by technology hardening of memory cells [4] and another one is by designing circuits able to detect an SEU event and act accordingly to prevent error propagation and guarantee full reliability in the system. Triple Modular Redundancy (TMR) or Error Detection and Correction Code (EDAC), like Hamming Code, are examples of such methods.

Filters are commonly used in digital communication systems for equalization, signal separation, noise reduction, etc. As communications are fundamental for space borne applications, such as satellites, unmanned missions, etc., digital filters play an important role in space systems [5].

There are two main types of digital filters: the recursive and the non-recursive filters. They are referred as infinite impulse response (IIR) filters and finite impulse response (FIR) filters, respectively. The FIR filters are widely used, as they have good stability and can be easily designed to match a given response.

This paper introduces optimizations for the use of Hamming Codes to protect generic (FIR) filter, described by (1) and illustrated in Figure 1, to show that by knowing attributes of the design to protect, it is possible to reduce resource consumption and achieve an optimal design.

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

As mentioned, TMR and Hamming EDAC are successful methods for protecting a design from SEUs. These techniques were studied by Hentschke *et al* in [6] for FIR filters.

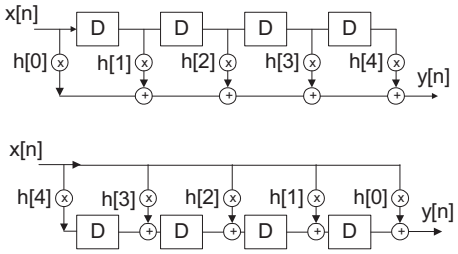

Figure 1: Direct Form FIR filter structure and transposed FIR Filter structure.

TMR is the most simple and effective way for protecting a design. The area consumption of this method is obviously up to three times higher, depending on which implementation is chosen. The optimal design of the TMR logic is discussed in [7], where Lima *et al* researched TMR logic implementations of a digital FIR filter for FPGAs.

Hamming codes are a simple class of block codes using the Hamming rule to determine, based on the number of information bits, the parity bits. This rule is articulated by the inequality given in (2)

$$d + p + 1 \leq 2^p \quad (2)$$

where d is the number of data bits and p is the number of parity bits. These codes have a minimum distance of three, and thus they are capable of correcting all single errors or detecting all combinations of two or fewer errors within a block (SEC-DED). Syndrome decoding is especially suited for Hamming codes. In fact, the syndrome can be formed to act as a binary pointer to identify the error location.

Hamming encoders and decoders perform specific combinational operations on the data in order to generate parity bits (in the case of encoders) and to correct errors (in the case of decoders).

The previous approach [6] to protect FIR filters using Hamming codes consists in adding one encoder and one

This work was supported by the Spanish Ministry of Science and Education under Grant ESP-2006-04163, the Regional Government of Madrid and the European Union FEDER programme.

S. Liu, P. Reviriego and J.A. Maestro are with Universidad Antonio de Nebrija, C/Pirineos, 55 E-28040 Madrid, Spain (phone: +34 914521100; fax: +34 914521110; email: {sliu, previrie, jmaestro}@nebrija.es).

decoder before and after each register that is going to be protected, as illustrated in Figure 2.

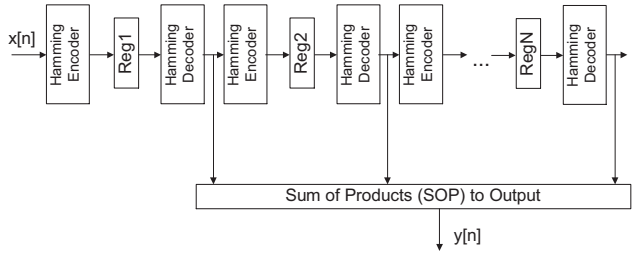

Figure 2: Existing Hamming EDAC protection approach.

This way of using EDAC codes to protect the circuit incurs a bigger delay in the critical path with respect to the use of TMR, what will be discussed further in this paper.
The area is obviously larger than in the case of the unprotected FIR, as *p* additional registers are needed in each tap plus the combinational logic for the encoders and decoders, but for some implementations it is lower than in the case of TMR, as shown in the following sections.

In the rest of the paper, alternative approaches to protect FIR filters with Hamming codes are presented.

II. PROPOSED TECHIQUES

In this section, several enhancements are proposed to reduce the number of encoders. These are based on the specific system knowledge of the FIR implementation, an approach that has been previously used to protect other circuits [8][9].

A. Hamming Single Encoder

One of these enhancements would be to remove the encoders from the delay line, as shown in Figure 3, as they are only used if there are errors in the circuit and even in that case, if only a single error is present in the register, it can still be corrected with the decoder at each stage. In summary, these additional encoders are useful only if we assume that a tap value will be hit by more than one SEU at different time instants, as it propagates through the delay line.

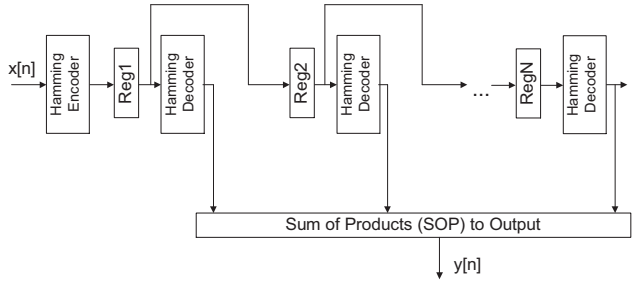

Figure 3: FIR filter protection with a single encoder.

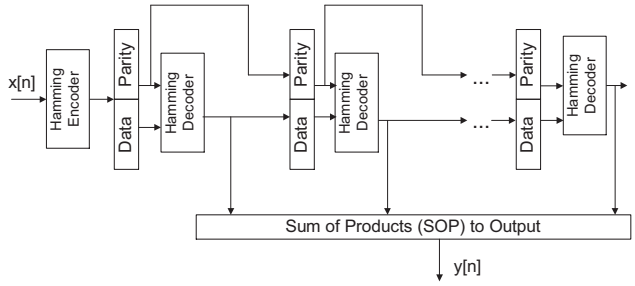

Figure 4: FIR filter protection with additional data protection.

B. Additional Hamming Data Protection

A further improvement would be to use the output of each decoder to feed the data bits of the next register while the parity bits are taken directly from the previous stage, as shown in Figure 4. This would allow recovering from multiple errors that occur in the data bits as long as they happen in different clock cycles. This is achieved without additional encoders.

C. Shared Hamming Decoder

A more sophisticated approach to reduce the complexity is shown in Figure 5, where the Hamming decoder is broken apart yielding a syndrome calculator, an error locator and an error corrector. The syndrome calculator retrieves through XOR operations of the data bits and the parity bits the syndrome (Figure 5.a). This should contain only zeros when there are no SEUs. When the data bits contain an SEU, then there will be 1's in the syndrome for identifying the SEU position in the locator, using the syndrome information and assuming that the SEU affects only one register.

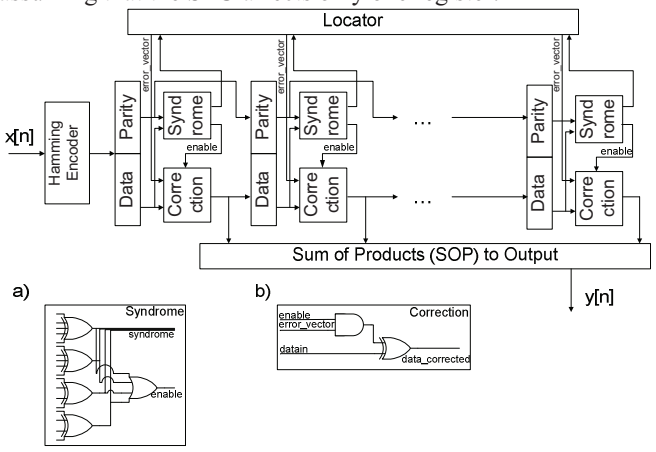

Figure 5: FIR with Hamming using one encoder and a common decoder; (a) Syndrome circuit, (b) Correction circuit.

The locator sends out an error vector with the exact position of the bit-flip to the corrector. This uses the OR-combined syndrome as an enable to initiate with the received error vector the correction of the faulty bit (Figure 5.b).

In this way, some of the logic (locator) is shared among all taps reducing overall complexity under the assumption that only one SEU per cycle will occur. Moreover, it allows recovering from multiple errors in the data bits in different clock cycles, as in the design of Figure 4. These improvements to the previous technique proposed in [6] are further researched in this paper to prove that they reduce the area cost.

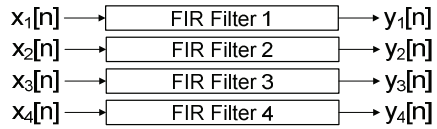

Figure 6: Four Parallel filters.

D. Exploiting Hamming through Parallelism

Nowadays, it is common to find systems where parallel FIR filter structures are used, as for instance in digital communication such as wireless LAN [10] or Ethernet [11],

where there are 2 or 4 parallel FIR filters, respectively (see Figure 6).

By unifying the data streams from all the channels and coding them as one, as depicted in Figure 7, reduction is achieved in the parity bits due to the fact that when doubling the data bits *d*, the parity bits *p* only increase by one. According to (2), when increasing the data bits from 8 to 16 or 32, the parity bits increase by 1 or 2, respectively. In such a case, it is possible to use the error vector from Figure 5 instead of doubling or quadrupling it for 2 or 4 parallel FIR filters, respectively. This is done by using different enables for each data block.

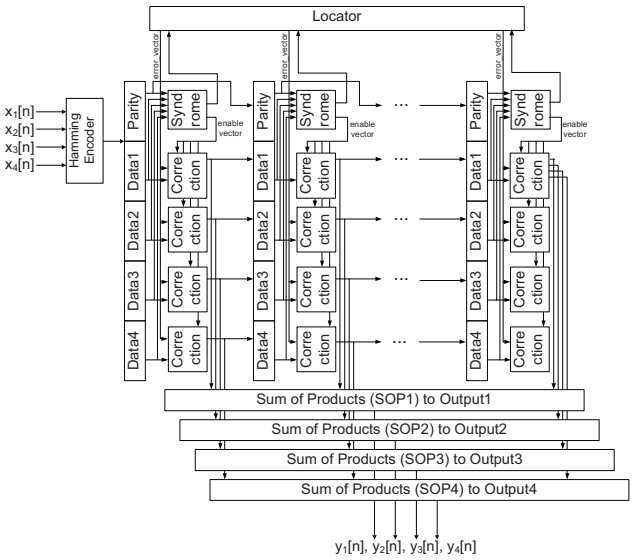

Figure 7: Four parallel FIR filter with Hamming protection.

All the techniques proposed in this section optimize and highlight the benefits of Hamming protection. However, Hamming codes have some drawbacks when they are compared with TMR. The main one is that for all proposed techniques, the decoder unit or the syndrome, locator and corrector are added to the critical path of the output, and therefore it decreases the maximum operational frequency of the circuit.

III. Experimental results

In this section, their effectiveness for actual implementations will be evaluated. To that end, the proposed techniques have been implemented in VHDL and then the circuits have been synthesized for a commercial ASIC library and for the Xilinx VIRTEX-5. This will also allow assessing the efficiency of the proposed techniques in terms of circuit complexity and compare it with the traditional approaches (like TMR). Then, an experimental setup [12] based on the Single Event Upset Simulation Tool (SST) has been used to insert SEUs.

The generic FIR filters have been implemented with 5 and 11 taps using the same structure as in [6] and [7], respectively. Equations (3) and (4) show the FIR filter coefficients, as mentioned in the references above.

$$h[n] = [-1 \;\; 24 \;\; 50 \;\; 50 \;\; 24 \;\; -1] \quad (3)$$

$$h[n] = [1 \;\; -1 \;\; -9 \;\; 6 \;\; 73 \;\; 120 \;\; 120 \;\; 73 \;\; 6 \;\; -9 \;\; -1 \;\; 1] \quad (4)$$

For SRAM-based FPGA, implementation errors could also occur in the configuration bits, what would cause changes in the circuit functionality [7]. The rest of the paper focuses on errors on the design flip-flop only and assumes that the configuration memory is protected by other techniques such as scrubbing [7].

A. Effectiveness

The results show that the effectiveness of all the compared techniques is similar, since they protect against isolated single events. However, when dealing with SEUs that occur a few clock cycles apart, the different techniques offer somewhat different behaviors. In the traditional Hamming protection, as long as there is only one single event per cycle, there will be no errors in the output. However, for the protection technique depicted in Figure 3, only one SEU could occur in each coded word, as it traverses the delay line. For example, if one bit-flip happens in the coded word of the second tap, and two cycles later (when the erroneous word reaches the fourth tap) another SEU occurs in one register of that fourth tap, the error will never be successfully corrected. Finally, the remaining proposed techniques will operate without errors even if single events occur in the data bits at each clock cycle. But if one SEU hits the parity bits of the coded word stored in the first tap, and during the next cycle another SEU occurs in the data bits of the second stage, the error will not be corrected. Nevertheless, in most cases, the probability of occurrence of single events a few clock cycles apart is negligible and therefore the protection provided by the proposed alternatives is similar to the traditional Hamming technique.

B. Complexity

The area cost (in equivalent gates and in VIRTEX-5 slices) of all the proposed protection techniques for 5 taps and 11 taps are shown in Table I for ASIC synthesis, where the results can be compared with the plain version of the FIR filter (unprotected) and with the TMR one.

TABLE I: ASIC SYNTHESIS RESULTS FOR ALL TECHNIQUES (8 BITS) WITH PERCENTAGE OF AREA COMPARED TO HAMMING

ASIC - TSMC 0,25 µm	FIR Filter 5 Taps			FIR Filter 11 Taps		
	Clk [MHz]	Gates	%	Clk [MHz]	Gates	%
without protection	144.7	755	-45.25	109.1	1,915	-41.67
TMR	126.9	1,545	12.04	99.9	3,549	8.10
Hamming	120.6	1,379	---------	92.0	3,283	---------
Hamming single encoder	116.5	1,293	-6.24	92.7	3,014	-8.19
Hamming additional data protection	118.6	1,266	-8.19	90.1	3,038	-7.46
Hamming shared decoder	113.8	1,263	-8.41	88.9	2,940	-10.45

As it can be seen when comparing the Hamming-based techniques, the one using split decoder units (Figure 5) offers the most competitive results, considering the combination of protection effectiveness and the total number of gates. The calculation of the area relation in percentage has been done in reference to the standard Hamming implementation. It can been seen that for 5 taps the *Hamming shared decoder* version uses 8.41% less area compared to the standard Hamming version and 20.41% compared to the TMR version. This reduction is even larger when increasing the filter taps up to 11 with 10.45% compared to standard Hamming but reduced to 18.10% compared to TMR.

Table II shows the FPGA synthesis results for all techniques, highlighting that the *Hamming shared decoder* gives the best performance in terms of LUT usage, as the FFs are determined

by the delay line and the extra parity bits. Comparing the synthesis results to that one of the ordinary Hamming, it achieves overall LUT savings of 12.02% for 5 taps and a 10.76% for 11 taps.

TABLE II: FPGA SYNTHESIS RESULTS FOR ALL TECHNIQUES

FPGA - VIRTEX 5	FIR Filter 5 Taps				FIR Filter 11 Taps			
	Slices	FF	LUT	Clk [MHz]	Slices	FF	LUT	Clk [MHz]
without protection	33	48	105	382.85	83	96	263	371.01
TMR	77	144	238	365.10	195	288	538	354.80
Hamming	59	72	208	352.61	147	144	474	346.14
Hamming single encoder	57	72	178	338.41	132	144	405	329.28
Hamming additional data protection	53	72	188	341.30	129	144	430	320.45
Hamming shared decoder	52	72	183	327.33	125	144	423	220.65

However, the operation frequency of the FIR filter protected using TMR is better because of the reduced combinational logic added to the critical path. Results of the synthesis for the parallel FIR filters with Hamming protection and the proposed techniques have been gathered in Table III for ASIC synthesis and Table IV for FPGA synthesis.

TABLE III: ASIC SYNTHESIS RESULTS WITH PERCENTAGE OF AREA COMPARED TO HAMMING FOR 8 BITS, 16 BITS AND 32 BITS.

ASIC - TSMC 0,25 µm		FIR Filter 5 Taps			Fir Filter 11 Taps		
		Clk [MHz]	Gates	%	Clk [MHz]	Gates	%
8 bits	Hamming	120.6	1,379	--------	92.0	3,283	--------
	TMR	136.0	1,487	7.8	99.9	3,549	8.1
	shared decoder	113.8	1,263	-8.4	89.9	2,912	-11.3
16 bits	Hamming	108.2	2,784	--------	92.9	6,556	--------
	TMR	126.0	3,104	11.5	81.7	7,073	7.9
	shared decoder	101.4	2,492	-10.5	78.6	5,851	-10.8
32 bits	Hamming	103.8	5,494	--------	78.9	12,882	--------
	TMR	128.7	6,204	12.9	92.9	14,146	9.8
	shared decoder	100.8	4,711	-14.3	72.0	11,076	-14.0

In the ASIC area results, a pattern of complexity reduction can be identified when increasing the data bits from 8 to 16 or 32 respectively, between ordinary Hamming and TMR. This raise is even more significant when comparing the results with that one of the Hamming shared decoder. In the case of 32 data bits, the difference between the proposed technique and ordinary Hamming is 14.3% and 27.2% comparing against TMR. The synthesis results of the 11-tap filter are similar. The savings of the proposed techniques are 14.0% and 23.8% against the ordinary Hamming and TMR, respectively. Furthermore, it can be observed that when increasing the data bits the reduction is significant, but when increasing the filter taps the reduction is smaller. This decrease is because the ordinary Hamming complexity is converging to TMR with the increase of the filter taps, as the percentage of the triplicated flip-flops in comparison to the combinational logic in the whole design is changing. This can be seen in the figures of the TMR overhead compared to ordinary Hamming of 12.9% for 5 taps and 9.8% for 11 taps, but the decrease from ordinary Hamming to the proposed technique from 14.3% to 14.0% is almost insignificant.

TABLE IV: FPGA SYNTHESIS RESULTS FOR 8 BITS, 16 BITS AND 32 BITS.

FPGA - VIRTEX 5		FIR Filter 5 Taps				FIR Filter 11 Taps			
		Slices	FF	LUT	Clk [MHz]	Slices	FF	LUT	Clk [MHz]
8bits	Hamming	59	72	208	352.61	147	144	474	346.14
	TMR	77	144	238	365.10	195	288	535	244.80
	shared decoder	52	72	183	327.33	125	144	392	220.65
16bits	Hamming	137	126	450	222.47	306	252	1020	185.12
	TMR	234	288	516	363.65	512	576	1165	240.73
	shared decoder	108	126	383	202.72	278	252	866	184.09
32bits	Hamming	260	228	897	183.39	587	456	1987	189.47
	TMR	456	576	1023	364.23	929	1152	2308	218.91
	shared decoder	229	228	773	176.37	539	456	1775	138.24

The VIRTEX results are as competitive as the ones discussed before for Table II. The decrement in the usage of LUTs when comparing the proposed technique to the ordinary Hamming is growing proportionally with the increase of the data bits, as seen in Table IV, where it changes for 5 taps from 14.89% to 13.83% (for 8 and 32 data bits respectively) and for 11 taps from 15.10% to 10.67% (for 8 and 32 data bits), respectively. The decrement of the results for the 32-bit version of the 5 taps or the 11 taps is in line with the results obtained and discussed for the ASIC results.

IV. CONCLUSIONS

Different approaches to protect generic FIR filters using Hamming codes against single events have been presented and put in perspective for ASICs and FPGAs. It has been shown that through understanding the system, enhancements can be carried out without losing any functionality. The reduction achieved for the ASIC implementation is 8.4% and 11.3% for 5 taps and 11 taps, respectively with 8 data bits. When increasing the data bits up to 32 the reduction is 14.3% and 14.0% for 5 taps and 11 taps, respectively.

Future work includes the research of FPGA oriented solutions for fault tolerant digital filters and the consideration of power consumption as a metric for optimization, since it is a key factor in space applications.

REFERENCES

[1] A.H. Johnston, "Scaling and Technology Issues for Soft Error Rates," *Proc. Fourth Ann. Research Conf. Reliability,* October 2000.

[2] R. D. Schrimpf, D. M. Fleetwood, "Radiation effects and soft errors in integrated circuits and electronic devices", World Scientific Publishing, 2004.

[3] P. E. Dodd, LL. Massengill, "Basic Mechanisms and Modeling of Single-Event Upset in Digital Microelectronics", IEEE Trans. on Nuclear Science, Vol. 50, No 3, June 2003, pp. 583 – 602.

[4] M. P. Baze, S. P. Buchner and D. Mcmorrow, "A Digital CMOS Technique for SEU Hardening", IEEE Transactions on Nuclear Science, Vol. 47, Issue 6, December 2000, pp. 2603 – 2608.

[5] S.M. Parkes, "DSP (Demanding Space-based Processing!): the Path Behind and the Road Ahead", 6th International Workshop on Digital Signal Processing Techniques for Space Applications, Noordwijk, The Netherlands, September 1998.

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

[7] F. Lima Kastensmidt, L, Sterpone, L. Carro and M. Sonza Reorda, " On the Optimal Design of Triple Modular Redundancy Logic for SRAM-based FPGAs", Proc. of the conference on Design, Automation and Test in Europe, Vol. 2, 2005, pp. 1290 – 1295.

[8] P. Reviriego, J.A. Maestro, O. Ruano, "Efficient Protection Techniques Against SEUs for Adaptive Filters: An Echo Canceller Case Study", IEEE Transactions on Nuclear Science, Vol. 55, No 3, June 2008, pp. 1700-1707.

[9] B. Shim, N.R Shanbhag and S. Lee, "Energy-efficient soft error-tolerant digital signal processing", Signals, Systems and Computers Conference Record of the Thirty-Seventh Asilomar, 2003, pp. 1493 – 1497.

[10] E. Perahia, "IEEE 802.11n Development: History, Process, and Technology", IEEE Communications, Vol. 46, No. 7, July 2008.

[11] J. Huang and R.R. Spencer, "The Design of Analog Front Ends for 1000BASE-T Receivers", IEEE Transactions on Circuits and Systems – II: Analog and Digital Signal Processing, Vol. 50, No. 10, October 2003.

[12] O. Ruano, J.A. Maestro, P. Reviriego, "Performance Analysis and Improvements for a Simulation-based Fault Injection Platform", Proc. of IEEE International Symposium on Industrial Electronics (ISIE'08), Cambridge (U.K.), June 2008, pp. 2299-2304.