Diverse Double Modular Redundancy: A New Direction for Soft Error Detection and Correction

Authors: P. Reviriego, C.J. Bleakley and J.A. Maestro.

Abstract

Soft errors are becoming an important issue for deep submicron technologies. To protect circuits against soft errors, designers routinely introduce modular redundancy to detect and correct these errors. A commonly used technique, Double Modular Redundancy (DMR) involves duplication of the basic module. Conventionally, DMR only allows error detection since voting cannot be used to determine the module in error. Recently, however, it has been found that DMR can, for some circuits, be enhanced to provide soft error correction as well as detection. The general approach, DDMR (Diverse DMR), relies on introducing design diversity between the original and redundant modules so that they produce different error patterns when a soft error occurs. The module in error can be found by examining these patterns. Herein, the generalized approach is described. A number of techniques for producing diverse designs with distinct error patterns are identified and illustrated with examples. New DDMR solutions are presented and finally, the future direction of DDMR research is discussed, in particular its application to other types of circuits and its use to protect against Common Mode Faults.

Introduction

Radiation-induced soft errors have become a major concern in the design of deep submicron digital Integrated Circuits (ICs) [1]. A soft error occurring due to a high-energy particle hit manifests itself as a transient bit reversal in the logic value of a circuit node. The bit reversal arising can be stored in a memory cell in the circuit and can cause output errors many clock cycles after the soft error occurred. For large process geometry technologies, radiation-induced soft errors are rare and so design for reliability to soft errors was primarily limited to mission critical ICs in space applications. Modern ICs, with smaller process geometries and lower supply voltages, are, however, more susceptible to soft errors and reliability is becoming a concern, even in terrestrial applications [2]. Naturally, high volume terrestrial applications are more cost sensitive than space applications. Hence, new low cost techniques to enhance circuit reliability in the event of soft errors are needed.

Over the years, a range of techniques has been proposed protect circuits against soft errors [3]. Triple Modular Redundancy (TMR) uses three identical copies of the original module operating in parallel. An error is detected if the outputs of the modules differ. The error is corrected by voting, i.e. taking the most common output value as the correct result. In the case of soft errors, this approach is effective since, due to their low rate of occurrence, the probability of two soft errors striking the same module in the same clock cycle is negligible. The difficulty with TMR is that it more than triples the area and cost of the circuit. A well-known alternative is Double Modular Redundancy (DMR) whereby the original module is duplicated. This reduces cost and provides error detection but, conventionally, error correction is not possible since voting cannot determine which of the two modules is in error. Recently, the modular redundancy concept has been extended to provide error correction in certain cases by monitoring the distribution of the module output errors. Using one redundant module, Soft-NMR [4] can correct soft errors if they cause the module-in-error to output a value that is unlikely compared to the typical distribution of outputs. While low cost, the technique does not correct all single errors.

In conventional modular redundancy, the redundant modules are identical copies of the original. Previously, researchers have proposed the use of design diversity to detect Common

Mode Failures (CMFs) [5]. CMFs are circuit failures, e.g. a supply voltage drop, that affect all modules in the same way. If the modules are identical, the outputs, although erroneous, will be the same and the error will not be detected. Design diversity resolves this problem by using functionally equivalent but structurally different redundant implementations of the original module. Thus, when a CMF occurs, the module outputs differ and the error is detected.

Design diversity has been used in fault tolerant systems for many years. One of its applications is to protect against design errors in software [6]. In this case, a duplex system using two independent implementations is used and errors are detected when the outputs of the two implementations differ. Some studies suggest that spending effort on implementing design diversity in a software project may be more cost effective than allocating that effort to testing and validation [7]. For circuits, design diversity has been used to protect against Common Mode Failures (CMFs) [8]. CMFs can be caused by events that affect the whole circuit, such a voltage drop, but also by aging effects, such as Electro-Migration and Negative Bias Temperature Instability [9]. These effects are related to the circuit structure and the data patterns used. Therefore, two identical copies of a circuit are likely to have similar aging effects that can lead to CMFs. As technology scales these effects are becoming prominent and are not a reliability challenge. Design diversity in circuits is also used to enable detection of permanent faults when temporal redundancy is used [10]. This is done by performing the same operation twice using different logic units each time.

There are a number of alternatives for implementing design diversity in circuits. One option is to add it at a low level of abstraction, for example, during logic synthesis [5] or place and route of the circuit [9]. At the logic level, two implementations may be forced to use different gate types to ensure design diversity [9]. Alternatively, different structures may be employed at the architectural level or different computation methods at the algorithm level. These latter two approaches are especially attractive for circuits that are rich in algorithmic and structural transformations, such as signal processing circuits. Regardless of the type of diversity adopted, it is important to quantify the diversity that is achieved, as this indicates effectiveness against CMFs. Various methods for quantifying design diversity have been proposed in the literature [11].

Other work has combined design diversity with repair techniques based on evolutionary algorithms aimed repairing partially faulty modules [12]. The use of design diversity on mixed signal circuits was recently explored in [13]. The results showed good protection against some multiple faults.

In a number of recent publications, the authors of this paper have investigated the use of design diversity within DMR to provide low cost detection and correction of radiation-induced soft errors. The general principle of the approach is that the two modules are designed in such a way that they give different output error patterns when a single soft error hits. These error patterns can be detected as a pattern of mismatches between the module outputs. By recognizing these patterns, the module-in-error can be identified and the output from the other module used as the final, error protected output. We refer to this general approach as Diverse DMR (DDMR). The design challenge within DDMR relies in identifying pairs of module implementations that provide distinct error patterns. To date, we have proposed and evaluated DDMR solutions for a number of common Digital Signal Processing (DSP) modules. We have focused on DSP modules since they have well-known alternative implementations and because DSP logic often consumes significant circuit area and power, making low cost protection all the more important.

This paper explains the general DDMR concept, presents an overview of existing DDMR solutions for specific DSP circuits and introduces new DDMR solutions for arithmetic circuits. A summary of results is provided. Finally, directions for future research on DDMR are suggested and the paper is concluded.

Diverse Double Module Redundancy

Figure 1 illustrates the generalized DDMR approach. DDMR relies on using two different implementations of the original module, referred to herein as M_1 and M_2, operating in parallel. To allow error correction, a soft error occurring in either M_1 or M_2 must produce an error pattern at the module outputs, e_1 or e_2, which is distinct from the error pattern that would be produced by the other module if it were hit, i.e. $e_1 \neq e_2$. In other words, the error patterns produced by M_1 and M_2 must be mutually exclusive. For example, M_1 might be designed so that all possible single soft errors hits corrupt all module outputs, whereas M_2 might be designed so that all possible single soft errors hits only corrupt some of the outputs. Alternatively, for sequential circuits, the differences in the error patterns might be designed to appear over time. For example, M_1 might be designed so that all possible single soft errors occurring in M_1 corrupt the module output for two clock cycles, whereas an error occurring in M_2 might only alter the output for one clock cycle. In all cases, the error is detected by observing the differences, or mismatches, between the module outputs. The error is corrected by: analyzing the mismatch pattern, inferring the module-in-error and selecting the output from the other module as the final, protected output.

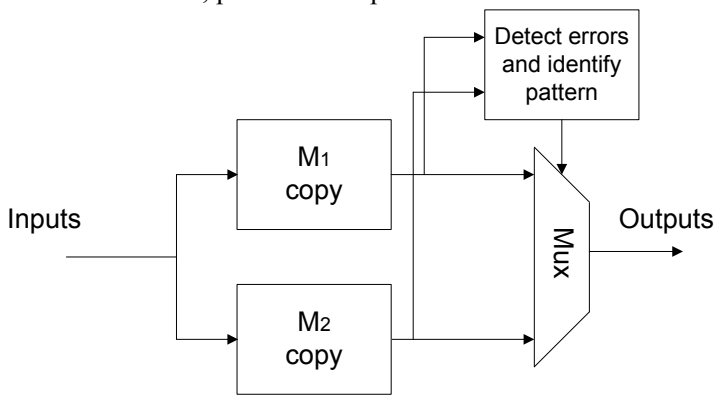

Figure 1. General DDMR approach.

More formally, we can say that the modules M_1 and M_2 perform functions m_1 and m_2 on the input data x to give unprotected intermediate outputs $y1$ and $y2$:

$$y_1 = m_1(x)$$
$$y_2 = m_2(x)$$

In the error-free case, $y_1 = y_2$ and either intermediate output can be presented as the final protected output. In the case of an error, $y_1 \neq y_2$ and an error discrimination function d must be applied to determine the module in error. The discrimination function must fulfil:

$$\begin{aligned} d(x, y_1, y_2) &= 0 \qquad \text{if } m_1 \text{ is in error} \\ &= 1 \qquad \text{if } m_2 \text{ is in error} \end{aligned}$$

It must be possible to determine the module in error based on the differences between y_1 and y_2 and optionally, their relationship to x. This requires that the error patterns arising from the modules are mutually exclusive, that is:

$$\forall i, j, x : m_1[i](x) \neq m_2[j](x)$$

where $m_k[i]$ denotes the logic function arising when module M_k suffers a soft error on circuit node i. The final protected output y is then obtained as:

$$
\begin{aligned}
y = y_1 \qquad & \text{if } (y_1 = y_2) \vee ((y_1 \neq y_2) \wedge (d = 1)) \\
= y_2 \qquad & \text{otherwise}
\end{aligned}
$$

The challenge in developing DDMR circuits lies in the fact that the designer must identify an efficient redundant implementation (M_2) that gives rise to an accurate and efficient discrimination function (d). Previous work on using different implementations of the same module has focused on protecting circuits against CMFs. In the case of CMF protection, the designer seeks to maximize a diversity metric that estimates the probability that when an error occurs in both modules (one in M_1 and the other in M_2), the modules produce different error patterns [5][11]. This can be done as part of the logic synthesis process [5]. However, for DDMR, the requirements are more stringent. Firstly, all possible soft errors must give rise to error patterns that are unique to one of the modules. Secondly, for cost reasons, it must be possible to distinguish the error patterns using reasonably simple logic circuits that only inspect the module outputs. Thirdly, again for cost reasons, the redundant module, while diverse, should not be significantly larger than the original module.
To the best of our knowledge, there is no single discrimination function (d) that meets these requirements for all possible modules (M_1). Hence, discrimination functions must be specifically designed for each module type. In the following section, we describe DDMR circuits for DSP and other applications.

DDMR Solutions for DSP circuits

One option for tackling the problem of finding appropriate diverse module implementations is to consider levels of abstraction above the logic level. This can be done for designs that have regular structural and/or algorithmic properties. These properties have been exploited for many years in the development of fault tolerant DSP circuits [14] since most DSP circuits have a large number of known alternative implementations [15] .
In [16], a DDMR solution was proposed for Finite Impulse Response (FIR) filters. After a soft error, a transpose FIR filter produces an output error in only one cycle, whereas, a functionally equivalent cascade of two FIR filters produces, in most cases, outputs several consecutive errors. The parts of the circuit that do not give rise to consecutive output errors can be protected using TMR, since they are small in area. The final scheme, named Structural DMR, is depicted in Figure 2. For the case study considered in [16] the cost was 2.25 times that of the unprotected FIR filter. The concept of mixed DDMR and TMR has the potential to extend the use of DDMR approach to a wide range of circuits.

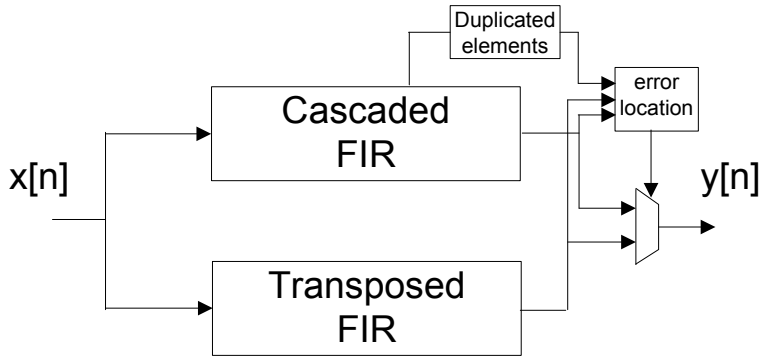

Figure 2. DDMR based on diverse structures applied to a FIR filter [16].

The use of completely different structures in each module is intuitively appealing. However, other techniques can be used. In some cases, a transform can be applied to the inputs of one module and undone at the output without affecting the final result. Input transformations such as this can be useful for DDMR if the transform alters the output error pattern in a detectable way. This technique was investigated for FIR filters in [17]. In so-called Signal Shaping

DMR, both modules use a transposed implementation filter. However, the input to one of the modules is transformed using a simple FIR filter. A corresponding inverse filter is applied at the output to cancel the effects of the input transformation. The scheme is shown in Figure 3. The added filter means that a single soft error causes multiple module output errors. In contrast, a single soft error in the conventional transpose module, without input transformation, only leads to a single output error. This Signal Shaping technique has the benefit that both modules are based on identical filter blocks. This facilitates implementation and ensures that performance is similar for both modules. The cost will be close to twice that of the unprotected filter as the added elements are much simpler than the protected filter.

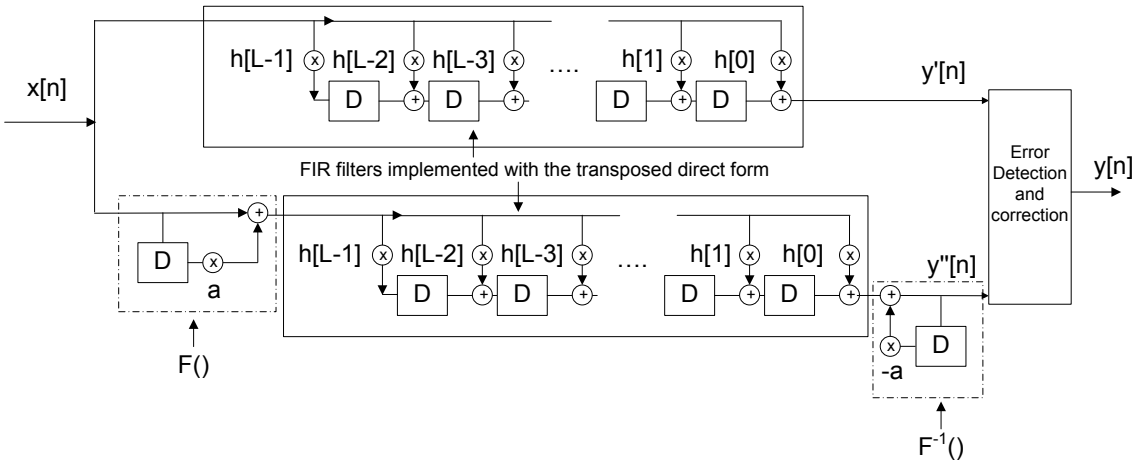

Figure 3. DDMR based on input-output transformation applied to a FIR filter [17].

In some cases, DDMR can be implemented using identical copies of a design with no modifications. This seems counterintuitive but can be done by ensuring diversity in the input data. For example, it is common in real-time DSP circuits for a stream of data to be processed in blocks. Diversity can be introduced by providing the original and redundant modules with different blocks of data. A simple way to do this is to time-shift the data blocks relative to one another. Depending on the processing in the module, a single soft error will typically affect multiple output samples in the same block. The block in error can be identified as the one containing all of the output mismatches. A version of this scheme, named Offset DMR, was used in [18] for convolutions (i.e. filters) implemented using the Fast Fourier Transform (FFT). An illustration of fault identification using block processing with time-shift is provided in Figure 4. When a single soft error occurs in M_1, it propagates to four output samples in that block. Hence four mismatches are detected where the outputs of M_1 and M_2 differ. Given that soft errors are rare and transient, by Occam's Razor, it is much more likely that a single error occurred in M_1 than two consecutive errors affecting two outputs blocks occurred in M_2. Obviously, the approach requires that soft errors in a block propagate to output samples in the first and second halves of the block. This arises naturally in FFT-based convolutions, except for soft errors of the final stages IFFT. These stages must be protected with TMR [18]. The overall scheme is shown in Figure 5. For the block sizes and bit-widths considered in [18], the cost was in the worst case 2.13 times that of the unprotected implementation so that the cost of DDMR is close to that of DMR.

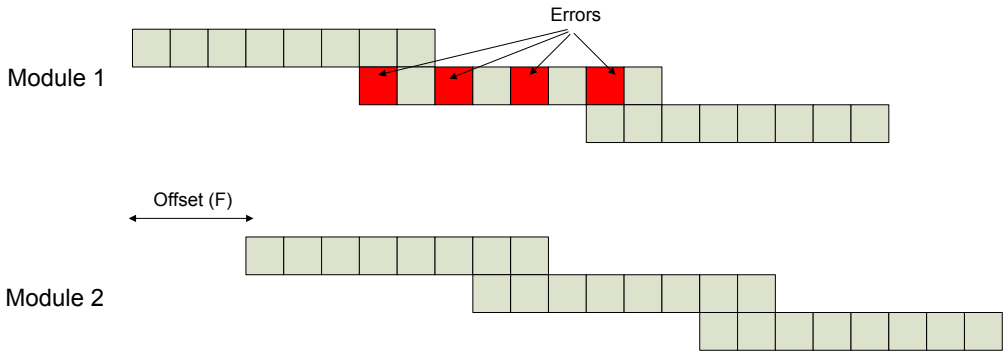

Figure 4. Identification of the copy in error using blocks and a shifted input in one of the copies [18].

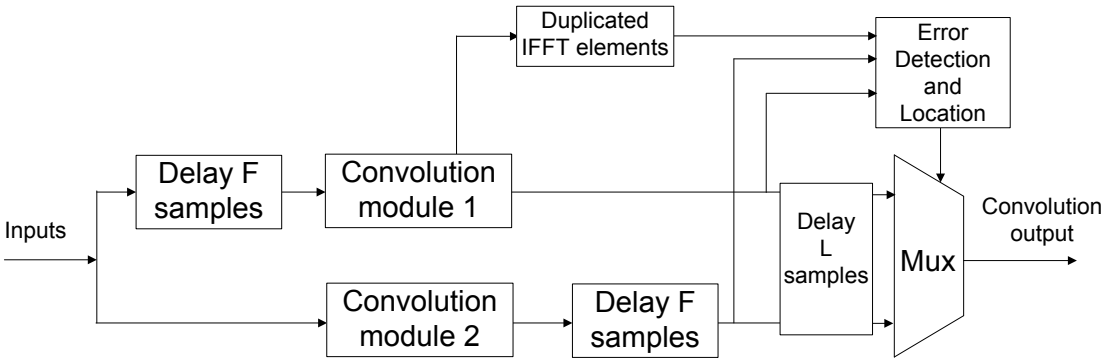

Figure 5. DDMR based on diverse input data applied to FFT-based convolution [18].

Of course, as well as the modules being susceptible to soft errors, the error detection and correction circuitry can also experience soft errors. An interesting observation is that, as can be seen in Figure 1, the error detection and correction logic essentially just selects one of the module outputs as the final output. This means that a soft error will only alter which module is selected for output. Given that soft error events typically occur singly, if a soft error occurs in the error detection and correction logic, it is likely that both module outputs are correct and the erroneous output selection does not affect the final output. This assumption holds, provided that the error detection and correction circuit is purely feed-forward and errors flush out of it quickly. This observation greatly simplifies protection of the error detection and correction logic.

It should be noted that the use of input data or implementation diversity in DDMR can lead to slightly different module outputs even in the error-free case. Thus, the error detection logic must be designed to tolerate small differences between module outputs. As a consequence, small output errors can go undetected. This is typically not an issue for DSP circuits since input and truncation noise are inevitable and systems are designed to be robust to small errors. All of the DDMR solutions described in this section have been implemented and tested in Matlab using random soft error injection campaigns. Some solutions have also been implemented in Verilog and verified by random soft error injection using the SST tool [12]. In the case of full-precision arithmetic, 100% of the errors are detected and corrected. In the case of reduced-precision (fixed-point) arithmetic, small sub-threshold errors are not detected. These errors are small in magnitude and have little or no effect since DSP systems are designed to be robust to input and quantization noise. The computational complexity or area of the solutions implemented was found to be in the range 205% - 225% of the basic module. In other words, close to conventional DMR. Since the results are parameter dependent, the reader is referred to the original papers for more details.

DDMR Solutions for Arithmetic Circuits

Arithmetic circuits are also suitable for DDMR. In the following, we propose two new DDMR solutions.

Consider the case in which a series of matrices, B, C, D, are multiplied by a single matrix, A. This occurs, for example, in computer graphics where matrix multiplication is used to perform rotation and scaling, of an object. DDMR can be implemented for a pair of matrix multiplications using architecture depicted in Figure 6. The basic module performs direct computation of the matrix multiplications. In contrast, the redundant module performs the matrix multiplication on the addition and subtraction of B and C. The addition/subtraction is undone at the output. This ensures that errors in matrix multiplication in the redundant module affect both $M'_b = A \times B$ and $M'_c = A \times C$. In contrast, errors in the basic module only affect M_b or M_c but not both. Therefore, error correction can be achieved by comparing M_b with M'_b and M_c with M'_c. If both are different then the error has occurred in the redundant module, if only one is different, then the error has occurred in the first module. The second module requires four matrix additions since dividing by two can be implemented with a simple right shift with no area cost. It is important to note that errors can also occur on those post-processing operations. Hence, they must be duplicated and protected separately. This means that six matrix additions are required. For $n \times n$ matrices, direct implementation of multiplication requires n^3 multiplications and n^3-n^2 additions, while addition requires n^2 additions. The cost of the unprotected implementation for computing M_b and M_c is $2n^3$ multiplications and $2(n^3-n^2)$ additions compared to the cost of DDMR, i.e. $4n^3$ multiplications and $4(n^3+0.5n^2)$ additions. This means that the cost of DDMR is close to two times that of the unprotected implementation. The proposed scheme has been implemented and tested in Matlab using random soft error injection campaigns.

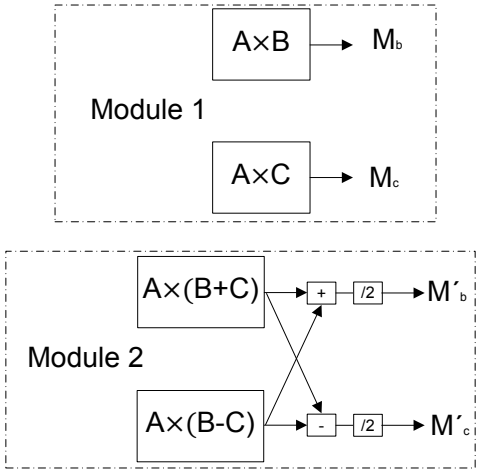

Figure 6. Modules to implement DDMR in matrix multiplications.

Another case in which DDMR can be used is in computation of the logarithm of a series of numbers, x, y, z. Consider computation of $\log(x)$ and $\log(y)$. DDMR can be applied by implementing the computation directly in the basic module and computing $a_1 = \log(x/y)$ and $a_2 = \log(x/y)$ followed by $x=(a_1+a_2)/2$ and $y=(a_1-a_2)/2$ by in the redundant module. As in the matrix multiplication case, errors in the logarithms of the basic module affect only one output while errors in the redundant module affect both outputs. This enables identification of the module in error and allows correction. The overhead required by DDMR depends on the relative cost of implementing a logarithm versus division, multiplication and addition. When the complexity of the first is much larger than the rest, the cost of DDMR is close to two times that of the unprotected implementation.

DDMR Future Directions

Table I provides a summary of the DDMR solutions found so far. We are currently working towards extending this solution space to other categories of circuits. From the examples of DDMR presented in the previous section, it is apparent that applying the technique to a given design is not straightforward. In some cases, a designer may leverage and extend existing DDMR techniques. For example, consider a circuit commonly found in communications ICs - a demodulator followed by an FIR filter where, to reduce area, the filter is implemented in the frequency domain using the FFT. Protection of the entire circuit can be achieved, as shown in Figure 7, by duplicating the modulator and applying Offset DMR to the filter.

A soft error in one of the modulators will corrupt only one sample of the input to corresponding filter. Therefore only one convolution output block will be affected at the output of that module. This ensures that the correction logic in Offset DMR detects and corrects the error. More generally speaking, this approach can be used for any modules that produce single output errors and operate consecutively with modules protected with DDMR such that the error patterns are a single error for one copy and multiple errors in the other copy. The technique has the additional advantage that the error detection and correction logic is shared between modules, further reducing the area cost of error protection.

Table I Summary of DDMR solutions.

Application	Diversity Technique
FIR Filter (Structural DMR)	Diverse structures
FIR Filter (Signal Shaping DMR)	Input-output transformation
FFT-based convolution (Offset DMR)	Diverse input data
Matrix multiplication	Sum-difference
Logarithmic processing	Sum-difference

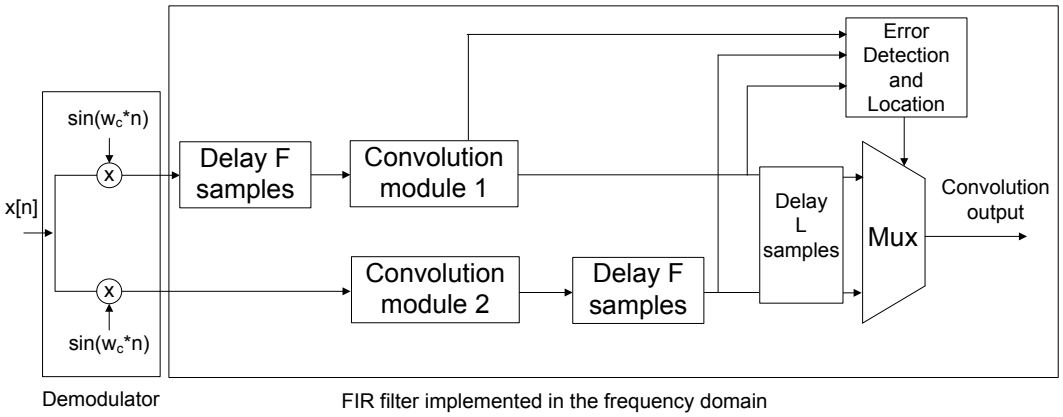

Figure 7. DDMR for a demodulator followed by an FIR filter.

Another potentially attractive application of DDMR is in protecting circuits against Common Mode Failures. Previously, design diversity was used to *prevent* output errors occurring in both modules simultaneously. With DDMR, design diversity can be exploited to *detect and correct* pairs of simultaneous *transient* errors (i.e. one error per module). For example, consider the DDMR protected filter shown in Figure 3. Recognizing the output error patterns and switching between the module outputs can correct most error pairs, excluding pairs that occur in the same tap in both modules (i.e., in the multipliers with the same delay), because the errors appear at the module outputs at different times. The basic DDMR concept can be enhanced for this application by correcting y''[n] when an error is detected. This CMF-enhanced DDMR concept was investigated in simulation by modeling an error protected filter and inserting pairs of errors, one per module, in randomly chosen multipliers. After fifty thousand error pairs, it was found that 78.9% and 88.3% of errors were corrected for an 8-tap

filter and a 16-tap filter, respectively. Thus multiple error correction is provided at very low cost.

Extending the use of DDMR to other types of circuits is an interesting avenue for future work. The techniques for matrix multiplications and logarithm computation presented in this paper show how it is possible to extend the use of DDMR beyond DSP circuits. When considering the use of DDMR, the ideal candidates are circuits rich in structure and/or regularity.

Conclusions

In this paper a new approach, Diverse Double Modular Redundancy (DDMR), has been proposed for the design of soft error tolerant circuits. For the circuits studied, DDMR detects and corrects almost all single soft errors at the cost of little more than doubling the circuit area. This is achieved by using diverse implementations of the module, or diverse input data transformations, such that, in the event of a soft error, the modules produce distinct error patterns at the module outputs. In this paper, the general approach has been described and several examples provided illustrating various diverse design techniques for DSP and arithmetic circuits. The techniques include the use of different circuit structures, different transformations of the input signal and different time-shifts for the data input blocks. The examples show the effectiveness of DDMR and motivate further research in DDMR. Finally, DDMR can also be used to protect against Common Mode Faults (CMFs). This concept is introduced and illustrated with an example that shows the efficiency of the technique compared to other previously proposed techniques.

References

[1] R. Baumann, "Soft errors in advanced computer systems", IEEE Design & Test of Computers, vol. 22, no. 3, pp. 258–266, May - June 2005.

[2] J. Maiz and N. Seifert, “Introduction to the Special Issue on Soft Errors and Data Integrity in Terrestrial Computer Systems” IEEE Transactions on Device and Materials Reliability, vol. 5, no 3, pp 303-304, September 2005.

[3] M. Nicolaidis, “Design for soft error mitigation,” IEEE Transactions on Device and Materials Reliability, vol. 5, no. 3, pp. 405–418, September 2005.

[4] E. P. Kim and N.R. Shanbhag, “Soft N-Modular Redundancy”, IEEE Transactions on Computers, vol. 60, no. 3, 2012, pp. 323-336, March 2012.

[5] S. Mitra and E. J. McCluskey, “Combinational logic synthesis for diversity in duplex systems” International Test Conference, pp. 179-188, 2000.

[6] A. Avizienis and J.P.J. Kelly, “Fault Tolerance by Design Diversity: Concepts and Experiments” IEEE Computer ,vol. 17 , no. 8, pp. 67 – 80, August 1984.

[7] K. Kanoun, “Real-world design diversity: a case study on cost”, IEEE Software, vol. 18, no. 4, pp. 29-33, July/August 2001.

[8] S. Mitra, N.R. Saxena and E.J. McCluskey, “Common-mode failures in redundant VLSI systems: a survey”, IEEE Transactions on Reliability, vol. 49, no. 3, pp. 285 – 295, September 2000.

[9] R. A. Ashraf, O. Mouri, R. Jadaa and R.F. Demara, “Design-for-Diversity for Improved Fault-Tolerance of TMR Systems on FPGAs” International Conference on Reconfigurable Computing and FPGAs, pp. 99 – 104, 2011.

[10] K. Wu and R. Karri, “Algorithm level recomputing using allocation diversity: a register transfer level approach to time redundancy-based concurrent error detection”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 21, no. 9, pp. 1077-1087, September 2002.

[11] S. Mitra, N.R. Saxena and E.J. McCluskey, "A design diversity metric and analysis of redundant systems", IEEE Transactions on Computers, vol. 51, no. 5, pp. 498-510, May 2002.

[12] S. Vigander, Evolutionary fault repair of electronics in space applications. Ph.D. Thesis, Dept. of Computer & Information Science, Norwegian University of Science and Technology (NTNU), Trondheim, 2001.

[13] G. d. M. Borges, L. F. Goncalves, T. R. Balen, and M. S. Lubaszewski, “Evaluating the effectiveness of a mixed-signal TMR scheme based on design diversity” In Proceedings of the 23rd symposium on Integrated circuits and system design, pp. 134–139,.

[14] A. Reddy and P. Banerjee, “Algorithm-based fault detection for signal processing applications,” IEEE Transaction on Computer*s*, vol. 39, no. 10, pp. 1304–1308, October 1990.

[15] A. V. Oppenheim and R. W. Schafer, Discrete Time Signal Processing. Englewood Cliffs, NJ: Prentice-Hall, 1999.

[16] P. Reviriego, C. Bleakley, J.A. Maestro, "Structural DMR: a Technique for Implementation of Soft Error Tolerant FIR Filters", IEEE Transactions on Circuits and Systems II, vol. 58, no 8, pp. 512-516, August 2011.

[17] P. Reviriego, C. Bleakley and J.A. Maestro, "Signal Shaping Dual Modular Redundancy for Soft Error Tolerant Finite Impulse Response Filters", IET Electronic Letters, vol. 47, no. 23, pp. 1272-1273, November 2011.
[18] P. Reviriego, C. Bleakley, J.A. Maestro, A. O'Donnell, "Offset DMR: A Low Overhead Soft Error Detection and Correction Technique for Transform Based Convolution", IEEE Transactions on Computers, vol. 60, no. 10, pp. 1511-1516, October 2011.
[19] D. Gonzalez, "Single event upset simulation tool functional description," ESA Rep. TEC-EDM/DCC-SST2, Jul. 2004.