Soft Error Detection and Correction for FFT Based Convolution using Different Block Lengths

Pedro Reviriego, Juan Antonio Maestro, Anne O'Donnell and Chris J. Bleakley

Abstract— The structure of radix-2 Fast Fourier Transforms of length 2^n where *n* is an integer is used to propose a new soft error detection and correction scheme for transform based convolution. The scheme can provide up to 100% detection and correction of isolated soft errors for, in many cases, approximately double the original system cost in terms of area and/or computational complexity. This is a substantial reduction when compared with conventional Triple Modular Redundancy. The method can be used for both hardware and software implementations of transform-based convolution.

Index Terms— Soft Error, FFT, Transform-based Convolution, Signal Processing.

I. INTRODUCTION

THE detection and correction of soft errors is an increasingly important problem in digital circuit design [1]. Soft errors arise from radiation particles that cause bit flips at the output of registers or combinational logic gates. For decades, soft errors have been a major concern for space applications due to circuit exposure to high radiation levels. With the introduction of smaller device geometries and new process technologies, circuits are more vulnerable to radiation and therefore soft errors are now becoming an issue for ground level applications as well [1]. In order to address the soft error issue, several methods have been proposed to protect circuits from the effects of soft errors. These range from the use of specific manufacturing techniques to the addition of redundancy at the system level to detect and correct the errors [2].

Convolution and filtering are among the most common functions in Digital Signal Processing systems. These functions can be efficiently implemented for long sequences [3] using transforms such as the Fast Fourier Transform (FFT). The main advantage of transform based convolution and filtering is that convolution is reduced to multiplication in the transform domain, significantly reducing computational complexity.

The use of convolution in many applications and the cost efficiency of transform based convolution implementations provide a strong motivation to study fault tolerant techniques for transform based convolutions. This paper introduces a new low overhead soft error detection and correction scheme for FFT-based convolution. The scheme uses a redundant convolution module that uses an FFT with different length from the one used in the main channel. When the outputs of both channels are different, the module in error is determined by analysis of the mismatch pattern at the module outputs. The scheme is designed for FFT Overlap Add convolutions with transform length 2^n where *n* is an integer. The scheme can be applied to hardware and software implementations. For large transform lengths, the area and computational complexity of the proposal tends to two-thirds of that of conventional Triple Modular Redundancy (TMR).

The remainder of the paper is structured as follows. Section II describes previously published research in the area of soft and permanent errors for transform-based convolution. Section III describes background theory on fast convolution by means of transforms. The proposed soft error detection and correction scheme is introduced in Section IV. Section V compares the area and computational complexity of the proposed scheme with that of TMR. Finally, a brief conclusion is given in Section VI.

II. RELATED WORK

The traditional approach for dealing with faults has been the use of Modular Redundancy (MR) that replicates the circuit such that errors can be detected when two modules are used and corrected when three identical modules are used. The first configuration is known as Dual Modular Redundancy (DMR) and the latter is known as Triple Modular Redundancy (TMR) and is widely used for fault tolerance [2]. An alternative approach is the use of Algorithm Based Fault Tolerance (ABFT) [4] in which fault tolerance is incorporated into the algorithm at the system level. ABFT has for example been applied to the computation of the Fast Fourier Transform [5]. Different ABFT approaches have been proposed for fault tolerant convolution. For example, in [6] cyclic error-correcting codes are used to implement a fault tolerant convolution. The proposed approach works for direct

This work was supported by the Spanish Ministry of Science and Education under Grant ESP-2006-04163.

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: {previrie, jmaestro}@nebrija.es).

A. O'Donnell and C J. Bleakley are with University College Dublin, Belfield, Dublin 4, Ireland (phone: +353-17165353; fax: +353-12695396; email: anneodonnell@eircom.net, chris.bleakley@ucd.ie).

978-1-4244-4595-0/09/$25.00

implementation of convolution but not for transform based convolutions. This is a major drawback as the cost of the direct implementation is, in most cases, much larger than that of a transform based implementation. In [7] an approach based on the use of Residue Number System (RNS) for the computation is presented. In this case, the implementation has the same underlying structure as that of transform based convolution and therefore the cost can be competitive. The results in [7] show an overhead of between 65% and 195% depending on the parameters of the convolution. These results compare favorably with the cost of TMR (200% overhead). More recently, alternative RNS approaches have been presented [8],[9] that relax some of the constraints imposed in [7] such as the need for co-prime moduli or the restrictions in transform length. Finally, as transform based convolution incorporates two transforms, methods for fault tolerant FFTs can be used. In [10] concurrent error detection is used in both transforms to provide fault tolerant convolution. However, the scheme only performs error detection. A number of alternative concurrent error detection schemes for FFTs have also been proposed (see for example [11],[12]) that could be used to protect convolutions in a similar way.

Most schemes proposed thus far address both permanent faults, including defects, and soft errors. Manufacturing defects cause continuous failure in the operation of some parts of the circuit significantly reducing yield and increasing the cost of good devices. Permanent faults also occur due to circuit degradation over time also causing some parts of the circuit to fail continuously. Dealing with permanent faults requires some degree of spatial redundancy. TMR and RNS based approaches offer spatial redundancy and therefore can deal with both permanent failures and soft errors. Unlike permanent faults, soft errors only cause temporary failure in some parts of the circuit. Thus, soft errors can be corrected using time redundancy so long as the duration of the malfunction is less than the correction time.

The scheme proposed in this paper uses a combination of spatial and temporal redundancy to detect and correct soft errors. The proposed approach is based on using two standard transform convolutions with different transform lengths in each channel. No sophisticated modification of the convolution implementation is needed. Correction can be performed without additional re-computation avoiding further delays. The overhead of the proposed scheme is, in many cases, close to 100%, which is competitive with previous methods, such as [7], especially given the simplicity of the proposed scheme.

III. TRANSFORM BASED CONVOLUTION

Direct computation of circular convolution is a very computationally expensive operation involving N^2 multiplications, where N is the length of the sequences to be convolved. The operation can be formulated as:

$$y(n \bmod N) = \sum_{k=0}^{N-1} h(k)\, x((n-k) \bmod N) \quad (1)$$

where *y(n)* is the circular convolution of the sequences *h(n)* and *x(n)*.

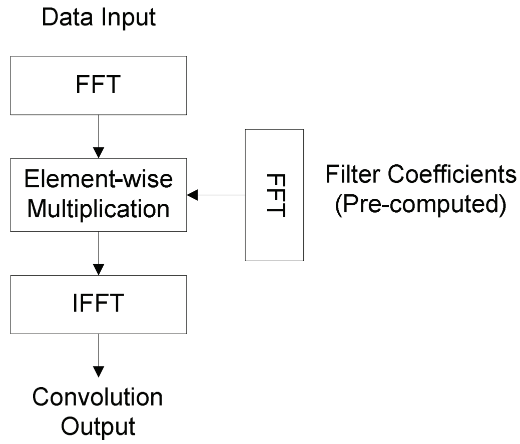

Fig. 1. Circular convolution using FFTs.

The convolution theorem for the Discrete Fourier Transform (DFT) indicates that the circular convolution of two finite sequences can be obtained as the inverse transform of the product of the transforms of the sequences [3]:

$$y(n \bmod N) = F^{-1}\left(F(h(n))\,F(x(n))\right) \quad (2)$$

where *F(x(n))* is the DFT of the sequence *x(n)*. The process of circular convolution using transforms is illustrated in Fig. 1 for a typical filtering application in which one of the sequences is fixed and pre-computed (the filter response). For sequences of length N where $N=2^n$ and n is an integer, the radix-2 Cooley-Tukey FFT algorithm may be used to reduce the computational complexity of the DFT and Inverse DFT to $N/2\log_2 N$ complex multiplies and $N\log_2 N$ complex additions [3]. This allows the calculation of the circular convolution in $N+3N\log_2 N$ complex operations, assuming that the second sequence is fixed and its DFT is pre-calculated as is normally the case in filtering applications. The FFT consists of n stages of calculation with $N/2$ butterflies in each stage where each butterfly operation is of the form:

$$y_0 = x_0 + x_1\omega^k$$
$$y_1 = x_0 - x_1\omega^k \quad (3)$$

For efficiency, the butterfly is implemented as a twiddle factor multiplication, $x_1\omega^k$, followed by a crossover (±).

In many practical linear filtering applications, one of the convolution input sequences is very long and the other, which is typically the impulse response of a filter, is comparatively short and fixed. When using transform-based methods, the long sequence is segmented into fixed size blocks prior to processing. The inputs and outputs of the circular convolution must be zero padded and overlapped to allow for the fact that the desired convolution is linear. There are two common methods - Overlap Save and Overlap Add. In this work, the Overlap Add method is used [3]. In this, the long sequence is broken into blocks of length L. These blocks are padded with M-1 zeroes where M is the length of the filter impulse response. The value of L is chosen such that the transform length $N=L+M-1$. The outputs of successive circular convolutions are overlapped by M-1 samples and the linear convolution is calculated by adding the samples in the overlap element-wise.

Use of the FFT to realise the transform can result in round-off error at the output of the convolution. The design has to ensure that the magnitude of those errors is small [11].

IV. Proposed Scheme

Herein, we propose that the convolution is performed by two modules, operating in parallel on the same input data. This is similar to DMR. However, we propose that two different FFT lengths are used in each module. We refer to this as FFT length DMR. The overall approach is illustrated in Fig. 2. The two modules shall be referred to as basic and redundant modules and they will perform independent computations of the convolution using different transform lengths N_b and N_r respectively.

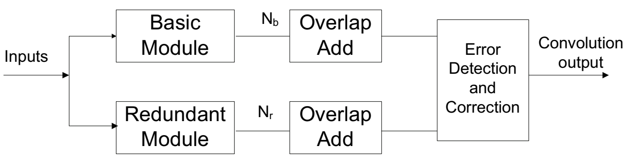

Fig. 2. Block Diagram of the proposed approach.

Since soft errors are rare events, we assume that only one soft error can occur in a short period of time i.e. consecutive convolutions blocks. As different block lengths are used in each module, the error pattern can be used to determine the module in error as explained below. Then correction can be performed by selecting the output from the other module.

A soft error in the forward FFT or in the multiplication stage will cause errors on all outputs of the convolution block. This is so because the errors will propagate to some of the inputs to the Inverse FFT and so, given the structure of the FFT/IFFT (shown in Fig. 3), will then propagate to all of the outputs of the Inverse FFT. As the two modules use different transforms lengths N_b for the basic module and N_r for the redundant one, if N_b differences are observed then the module in error is the first one. Conversely if N_r errors are observed then the soft error has occurred in the second module. This is illustrated in Fig. 4 where the error patterns at the output of the module for soft errors in the forward FFT or multiplication stage are shown for both the basic and the redundant channel. The samples for which a difference occurs are shown in red in both channels and errors are marked with a cross.

Errors in the inverse FFT (IFFT) will only cause differences in some of the outputs of the block but the error pattern can still be used to correct most of the errors. In general, for an FFT of length $N_i = 2^k$, a single error at the input to a stage g gives rise to r errors with a separation s at the output of the FFT where $r=N_i/2^{g-1}$ and $s=2^{g-1}$. As the basic and redundant channels have different values of N the error patterns will be different and can be used to locate the module in error and correct the error by selecting the output from the other module.

This is better explained with an example. Consider that two modules are used one with N_b=8 and the other with N_r=16 for a filter length M = 4. If a soft error occurs in the second stage of the Inverse FFT (IFFT) of the basic module with N_b=8 the soft error propagates to the outputs of the transform as shown in Fig. 3. In this case errors on $X(0)$,$X(2)$,$X(4)$ and $X(6)$ will be observed at the FFT output and then through the overlap add section. The four errors with a separation of two will reach the output of the basic module as illustrated in Fig. 5 on the upper part, again the samples for which a difference occurs are shown in red in both channels and errors are marked with a cross. For the redundant module, four errors will occur when g = 3 such that $r = N_r/2^{g-1}$ =4. In this case s= 4 as shown in the lower part of Fig. 5 and therefore the separation of the errors can be used to detect the module in error. Correction is then done by selecting the output from the other module.

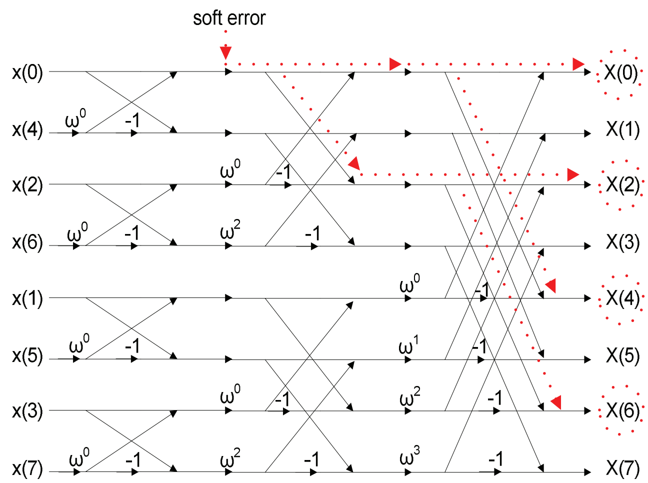

Fig. 3. 8-point radix-2, FFT showing path of error at the input to stage 2.

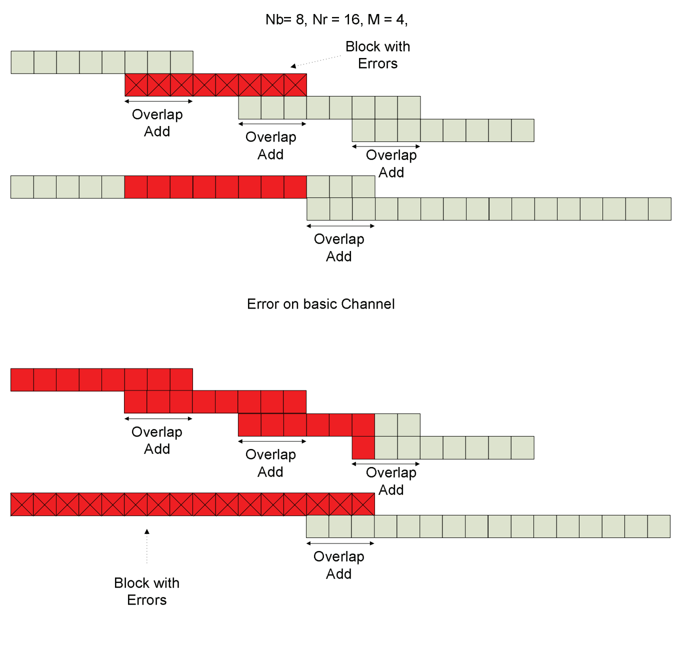

Fig. 4. Examples of output Errors patterns for errors on the forward FFT or multiplication stage for the basic (top) and redundant channel (bottom).

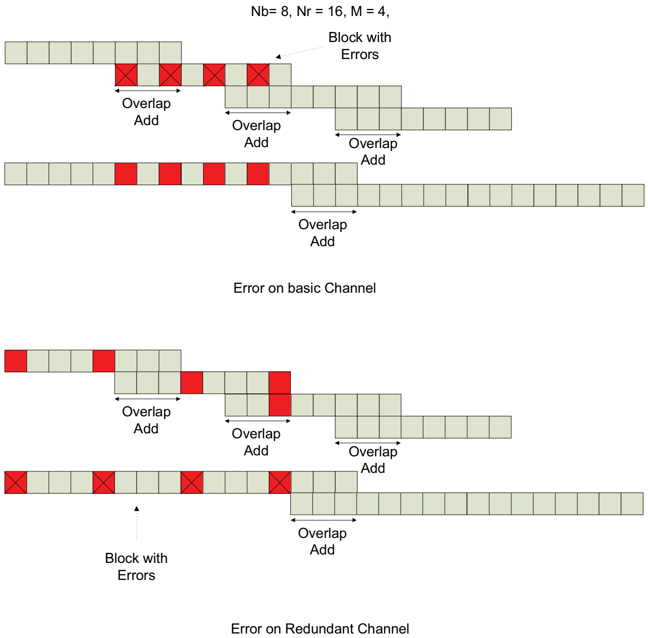

Fig. 5. Examples of output Errors patterns for errors on the IFFT of the basic (stage 2) and redundant channel (stage 3).

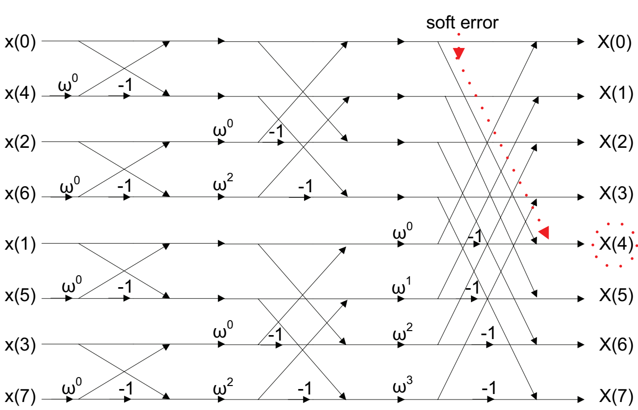

Fig. 6. Example of soft error at the last stage of the IFFT.

The exception is errors on the last stage of the IFFT for which $r = 1$ and therefore a single error is observed. This is illustrated in Fig. 6. Correction of those errors can be done by duplicating the last stage in one of the modules. Errors on the last stage of the module are easily detected by comparing the duplicated outputs. Errors on the last stage of the other module are detected when a single error is observed at the output and the duplicated outputs of the other module are equal. In both cases the errors can then be corrected by selecting the output from the other module.

When the transform lengths are given by $N_r = 2^* N_b$ and $N_b = 2^n$, a simplified procedure can be used to determine the module in error. In this case when there is an error at stage g in the redundant module we have $r=N_r/2^{g-1}$ errors at a separation of $s=2^{g-1}$ and then $N_b/s = r/2$ such that the position N_b points after the first error corresponds to the error $r/2$. Therefore, if the first observed error occurs at sample k then sample $k+N_b$ will also be in error if the error has occurred in the redundant module. If for sample $k+N_b$ the outputs from both modules are equal then the error occurred in the basic module otherwise the soft error has affected the redundant module. This greatly simplifies the correcting logic, as when a difference is observed only one additional output sample has to be checked to determine the module in error. The overall scheme is shown in Fig. 7 where the outputs of the modules are fed into delay lines of N_b samples and comparisons between modules are done at the input and output of the delay lines. If a difference is seen on both comparisons then the output from the basic channel is selected for the next N_r samples otherwise the output is taken from the redundant channel.

In practical systems, the FFT and IFFT use fixed-point arithmetic, leading to round-off error at the module outputs. Based on an analysis of the module, the round-off error can be bounded to be less than η [10]. Since the modules operate on different data blocks, the round-off error will be different in corresponding module outputs. Thus a mismatch is only deemed to have occurred in cases where the difference between corresponding outputs is greater than $2^*\eta$. These round-off errors may mask some of the errors produced by soft errors. It is likely that in most cases such masking occurs only for small errors and therefore with little impact on performance. If error masking becomes an issue then more robust techniques can be used to locate the module in error by using all observed output errors in the process. That would introduce additional complexity in the correction logic.

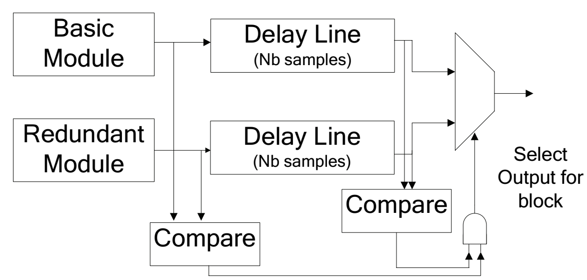

Fig. 7. Implementation of the proposed correction scheme.

V. Complexity Analysis

In this section, the complexity of the proposed scheme is compared with that of TMR in terms of the number of complex operations needed to implement both alternatives. TMR has been chosen as it is the traditional benchmark for comparison and because no detailed cost analysis is provided for most of the previously proposed techniques. Another important reason is that the proposed scheme is quite similar to TMR from an implementation point of view. Thus it is straightforward for a designer to add these techniques to an unprotected convolution implementation. Alternative approaches, such as the ones based on RNS, cyclic codes or concurrent error detection on the transforms require significant changes to the unprotected design and therefore involve a larger design effort than TMR or FFT length DMR. As a reference, the figures provided in [7] for the protection overhead lie between 65% and 195% compare to the values close to 100% typically achieved by the proposed scheme. This indicates that the proposed scheme can be competitive in terms of cost.

As already mentioned, circular convolution when implemented with FFT can be performed for a sequence of length N with $N+3N\log_2 N$ complex operations. Additionally for each block in the overlap method M-1 additions are needed and N-M+1 output samples are produced. Therefore the cost per output sample is given in terms of complex operations, by

$$Complexity = \frac{(N + 3N\log_2 N + M - 1)}{(N - M + 1)} \quad (4)$$

Therefore the cost for TMR is

$$Complexity_{TMR} = 3 \cdot \frac{(N + 3N\log_2 N + M - 1)}{(N - M + 1)} \quad (5)$$

And the cost for the proposed approach assuming $N_r = 2^* N_b$ is

$$Complexity_{proposed} = \frac{N_b + 3N_b\log_2 N_b + M - 1 + 1.5 * N_b}{(N_b - M + 1)} + \frac{2N_b + 6N_b\log_2(2N_b) + M - 1}{2N_b - M + 1} \quad (6)$$

where the additional 1.5*N_b operations are for the duplicated last stage in one of the modules (in this case in the module with length N_b). In the following, for simplicity, duplication of this last stage applied to the module with the smallest N. This does not necessarily result in the lowest complexity but its effect on the overall complexity for the examples presented here is negligible. In a given design it is straightforward to select the module that will provide the lowest cost for duplication of the last stage.

When $N >> M$ equations (5) and (6) lead to the approximation

$$Complexity_{proposed} = 2/3 * Complexity_{TMR.} \quad (7)$$

From (4), the optimum value of N can be selected for a given M. This is illustrated in Table I for two values of M. It can be observed that complexity is large for small values of N and then decreases to a value from which it starts to increase again.

TABLE I
COMPLEXITY ESTIMATES FOR FFT BASED CONVOLUTION FOR VARIOUS FILTER AND TRANSFORM LENGTHS.

N	M	Complexity
32	18	35,33
64	18	26,25
128	**18**	**25,53**
256	18	26,85
128	60	41,68
256	60	32,79
512	**60**	**31,77**
1024	60	32,95

Assuming that the optimum value of N is used in the implementation, the cost for TMR and for the proposed method can be calculated using (5) and (6). For the proposed method transform lengths of 2*N and N/2 can be used in the redundant module. In the latter case, it is straightforward to modify (6) for a transform length of N/2. The complexity estimates for the previous examples are shown in Table II. It can be seen that the savings are significant when compared to TMR and close to the 2/3 ratio predicted in (7) when $N >> M$. Additionally the results are similar when transform lengths 2*N or N/2 are used in the redundant module

TABLE II
COMPLEXITY ESTIMATES FOR PROPOSED METHOD AND TMR FOR VARIOUS FILTER AND TRANSFORM LENGTHS.

N	M	TMR	Proposed 2*N	Proposed N/2	Ratio 2*N	Ratio N/2
128	18	76,59	54,11	53,82	0,706	0,702
512	60	95,33	66,43	66,52	0,697	0,698

In some applications it may not be possible to use the optimum value for N due to, for example, latency constraints. Latency in transform based convolution increases with transform length. Therefore using large values of N implies a large latency. In some communications systems there are tight latency constraints, for example in the Ethernet standard for 1000Base-T a low value for the latency is specified [13]. Low latency is also required in some audio processing applications [14].

To illustrate the implications of latency constraints, in Table III the complexity estimates for the previous examples are presented but assuming that in the first case N has to be no greater than 64 and in the second case no greater than 256. The first thing to note is that since the values that N can take have to be smaller than the N used in the basic module now there is only one option for the redundant channel length: N/2. The second observation is that the complexity ratio deviates substantially from 2/3 implying that the complexity of the proposed method is closer to that of TMR. In fact there may be cases in which the proposed method results in a larger complexity than TMR. For example if for M = 60, latency constraints force values of N to be equal to or smaller than 128 then the redundant channel would have to use N = 64 and the complexity for the proposed method would be 299 complex operations compared to 125 for TMR. This is due to the fact that when N is close to M, complexity grows substantially as the denominator in (4) becomes small. From the example it becomes apparent that the proposed technique will be less cost effective or even more costly than TMR when there are latency constraints.

It should also be noted that the proposed technique adds additional latency, as buffering is required for error correction. This buffering as discussed before will add N/2+1 samples delay with N being the largest transform length used in the two modules. This further reduces the applicability of the proposed method for applications with tight latency constraints.

TABLE III

COMPLEXITY ESTIMATES FOR PROPOSED METHOD AND TMR FOR VARIOUS FILTER AND TRANSFORM LENGTHS WITH LATENCY CONSTRAINTS.

N	M	TMR	Proposed N/2	Ratio N/2
64	18	78,76	64,78	0,822
256	60	98,37	77,25	0,785

Finally it should be noted that the voting logic in TMR and the correction logic for the proposed approach have not been included in the complexity analysis. In both cases this logic is simple and its complexity should be negligible compared to the FFT, multiplication and IFFT calculations.

VI. CONCLUSIONS

A novel, low overhead soft error detection and correction technique for transform-based convolution has been introduced. For filtering applications, it has been shown that the percentage complexity of the proposed FFT length DMR compared to TMR tends to 66.6% as the transform length becomes large relative to the filter impulse response length. Another advantage of the proposed scheme is that it does not require the use of sophisticated arithmetic like RNS or complex coding of the inputs like many of the concurrent error detection approaches to protect transforms. This simplicity facilitates its applicability in real designs.

REFERENCES

[1] Baumann, R.; "Soft errors in advanced computer systems" IEEE Design & Test of Computers, vol. 22, Issue 3, May - June 2005 pp. 258 – 266.

[2] Nicolaidis, M., 'Design for soft error mitigation' IEEE Transactions on Device and Materials Reliability vol. 5, issue 3, Sept. 2005 pp. 405 – 418.

[3] Proakis, J.K., and Manolakis, D.G.; 'Digital Signal Processing: Principles, Algorithms, and Applications', 3rd Edition, Prentice Hall, 1996.

[4] Reddy A. and Banarjee P.; 'Algorithm-based fault detection for signal processing applications', IEEE Transactions on Computers vol. 39, issue 10, Oct. 1990, pp 1304 – 1308

[5] Jou J. and Abraham J.A.; 'Fault-tolerant FFT networks' Computers, IEEE Transactions on, vol. 37, issue 5, May 1988.

[6] Redinbo, G.R.; 'System level reliability in convolution computations' IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 37, issue 8, Aug. 1989 pp.1241 – 1252.

[7] Beckmann, P.E. and Musicus, B. R.; 'Fast fault-tolerant digital convolution using a polynomial residue number system' IEEE Transactions on Signal Processing vol. 41, issue 7, July 1993 pp. 2300 – 2313.

[8] Sundaram, S. and Hadjicostis, C.N.; 'Fault-Tolerant Convolution Via Chinese Remainder Codes Constructed From Non-Coprime Moduli' IEEE Transactions on Signal Processing, vol. 56, issue 9, Sept. 2008 pp:4244 – 4254.

[9] Conway, R.; 'Modified Overlap Technique Using Fermat and Mersenne Transforms', IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 53, Issue 8, Aug. 2006, pp. 632-636.

[10] Tahir, J.M., Dlay, S.S., Naguib, R.N.G. and Hinton O.R.; 'Self-checking architectures for fast Hartley transform' European Design and Test Conference,.6-9 March 1995 pp: 363 – 371.

[11] Choong G. O., Hee Y. and Raj V.K.; 'An efficient algorithm-based concurrent error detection for FFT networks', IEEE Transactions on Computers vol. 44, issue 9, Sept. 1995.

[12] Wang S. and Jha N.K.; 'Algorithm-based fault tolerance for FFT networks' IEEE Transactions on Computers, vol. 43, issue 7, July 1994.

[13] ´Phyiscal Layer Parameters and Specifications for 1000Mb/s operation over 4-Pair of Category 5 Balanced Copper Cabling Type 1000Base-T´ IEEE standard 802.3ab.

[14] Reviriego P., J. Parera and R. Garcia ´Linear-Phase Crossover Design Using Digital IIR Filters´ Journal of the Audio Engineering Society, vol 46, issue 5, May 1998, pp. 406-411.