Accumulation-Based Concurrent Fault Detection for Linear Digital State Variable Systems *

Ismet Bayraktaroglu
Computer Science & Engineering Department
University of California, San Diego
La Jolla, CA 92093
ibayrakt@cs.ucsd.edu

Alex Orailoglu
Computer Science & Engineering Department
University of California, San Diego
La Jolla, CA 92093
alex@cs.ucsd.edu

Abstract

An algorithmic fault detection scheme for linear digital state variable systems is proposed. The proposed scheme eliminates the necessity of observing the internal states of the system for concurrent fault detection by utilizing an accumulation-based approach. Observation merely of the inputs and the outputs results in significantly reduced area overhead and no performance penalty. Experimental results verify that 100% concurrent fault detection is attainable for linear digital state variable systems.

1. Introduction

With increasing levels of electronic penetration, it becomes all the more important to try to withstand possible hardware failures, lest they cascade throughout highly interconnected domains. Nonetheless, cost issues as always loom paramount. We propose a highly cost-effective approach for the concurrent detection of faults in linear digital systems. As DSPs constitute an integral part, if not the central locus, of a large number of electronic applications, a variety of concurrent error detection schemes, mostly reliant on algorithmic approaches, have been previously suggested, such as for digital filters [1], FFT networks [4, 7, 6], QR factorization [6] and linear digital systems [2].

The primary concurrent fault detection approach for the fundamental operations of digital systems, proposed by Huang and Abraham [3], provides error detection and correction for matrix operations through checksum codes. As noninteger implementations may cause individual checksum comparisons to fail due to rounding effects, real-number codes have been proposed as an alternative [5]. Utilizing the approaches proposed for embedding fault tolerance into matrix operators [3, 5], fault tolerant linear digital state variable systems have been subsequently outlined [2]. While high fault coverages are therein attained, access to internal states does increase the cost of error detection hardware appreciably. Recently, a time-extended invariant-based concurrent fault detection scheme has been proposed for FIR and IIR filters [1]. The time-extended nature of the invariant utilized obviates the internal state observation necessity of previously suggested schemes, thus appreciably reducing the hardware overhead.

In this work, we propose a concurrent error detection scheme for a general set of linear digital systems. The proposed scheme requires access solely to inputs and outputs of the system; a time-extended invariant is continuously checked. Even though lack of access to the internal states of the system introduces uncertainty to the invariant, high fault detection is achieved through accumulation of fault effects, nonetheless. Accumulation of fault effects results in an inverse correlation between fault magnitude and latency, a positive outcome as it implies that faults which may impact the system significantly are caught expeditiously. The superior fault detection capabilities of the scheme are shown analytically and are verified experimentally herein.

Section 2 defines the class of systems for which the proposed scheme is applicable. Derivation of the invariant and its hardware implementation is provided in section 3. While section 4 verifies analytically that the simple invariant checking mechanism indeed provides exceedingly high fault coverage, section 5 confirms the same fact experimentally. Section 6 discusses the significance of the results obtained in this work.

2. System Definition

While digital linear systems can be written as a set of equations in various ways, a state variable description is the most general representational form. A canonical form is given in equations 1 and 2, showing the next state and output functions, respectively; with n inputs, m outputs and k internal states; $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, and $\boldsymbol{D}$ denote matrices of dimensions $k \times k$, $k \times n$, $m \times k$, and $m \times n$, respectively.

$$\boldsymbol{s}^{t+1} = \boldsymbol{A}\boldsymbol{s}^{t} + \boldsymbol{B}\boldsymbol{u}^{t} \quad (1)$$

$$\boldsymbol{y}^{t} = \boldsymbol{C}\boldsymbol{s}^{t} + \boldsymbol{D}\boldsymbol{u}^{t} \quad (2)$$

*This work is being supported through a grant by Hughes Space and Communications and the UC Micro Program.

Throughout this paper, we use $\boldsymbol{s}$ to denote the internal states of the system, while $\boldsymbol{u}$ and $\boldsymbol{y}$ denote the inputs and the outputs of the system, respectively. Superscripts of column vectors denote a particular time instance, while bold capital letters are reserved for matrices and bold lowercase letters for column vectors.

3. Concurrent Test Implementation

Error detection schemes that access solely inputs and outputs result in low-cost implementations. We propose an accumulation-based approach, wherein the effect of the internal states in the long run is shown to be minimal, thus enabling their construal as part of a tolerance factor. On the other hand, all fault effects are guaranteed to accumulate over time to exceed this tolerance factor introduced through lack of state observation. Detailed derivation of the input/output relation that is independent of the internal states is provided in the following subsection, followed up with a subsection dedicated to proving an upper bound in the tolerance term of the invariant checking equation.

3.1. Derivation of the Invariant

Derivation of an invariant between the inputs and the outputs of a linear time invariant system requires a relation that is independent of the internal states. While the original output function depends both on the inputs and the internal states of the system, by solving the recursive next state equation (equation 1), we obtain:

$$\boldsymbol{s}^t = \boldsymbol{A}^t\boldsymbol{s}^0 + \sum_{k=0}^{t-1}\boldsymbol{A}^{t-1-k}\boldsymbol{B}\boldsymbol{u}^k \quad (3)$$

Equation 3 shows that, given the internal state of the system at time 0 and the set of inputs from time 0 to $t-1$, the internal state of the system at time t can easily be determined. By substituting the expression for $\boldsymbol{s}^t$ into equation 2, the result can be extended to the output at time t.

$$\boldsymbol{y}^t = \boldsymbol{C}\boldsymbol{A}^t\boldsymbol{s}^0 + \boldsymbol{C}\sum_{k=0}^{t-1}\boldsymbol{A}^{t-1-k}\boldsymbol{B}\boldsymbol{u}^k + \boldsymbol{D}\boldsymbol{u}^t \quad (4)$$

While equation 4, assuming that all initial states are zero, provides a relation between the inputs and outputs, checking this relation is essentially equivalent to replicating the original system. An alternative cost-effective relation needs to be attained to ensure viability of the scheme we propose. The desired complexity reduction of the relation in equation 4 can be achieved by taking the summation of both sides of the equation over the time range 1 to T and manipulating the resultant equation. In performing the summation, $\boldsymbol{y}^0$ is added to the left hand side of the equation and $\boldsymbol{C}\boldsymbol{s}^0 + \boldsymbol{D}\boldsymbol{u}^0$ is added to the right hand side of the equation.

$$\sum_{t=0}^{T}\boldsymbol{y}^t = \boldsymbol{C}\sum_{t=0}^{T}\boldsymbol{A}^t\boldsymbol{s}^0 + \boldsymbol{C}\sum_{t=1}^{T}\sum_{k=0}^{t-1}\boldsymbol{A}^{t-1-k}\boldsymbol{B}\boldsymbol{u}^k + \sum_{t=0}^{T}\boldsymbol{D}\boldsymbol{u}^t \quad (5)$$

By changing the order of summation for the second term of the right hand side of equation 5 and manipulating the summation index, we arrive at:

$$\sum_{t=0}^{T}\boldsymbol{y}^t = \boldsymbol{C}\sum_{t=0}^{T}\boldsymbol{A}^t\boldsymbol{s}^0 + \boldsymbol{C}\sum_{k=0}^{T-1}\sum_{m=0}^{T-k-1}\boldsymbol{A}^m\boldsymbol{B}\boldsymbol{u}^k + \sum_{t=0}^{T}\boldsymbol{D}\boldsymbol{u}^t \quad (6)$$

The second term can be split into 3 parts:

$$\begin{aligned}\sum_{t=0}^{T}\boldsymbol{y}^t &= \boldsymbol{C}\sum_{t=0}^{T}\boldsymbol{A}^t\boldsymbol{s}^0 + \sum_{t=0}^{T}\boldsymbol{D}\boldsymbol{u}^t + \boldsymbol{C}\sum_{k=0}^{T}\sum_{m=0}^{T}\boldsymbol{A}^m\boldsymbol{B}\boldsymbol{u}^k - \\ &\quad \boldsymbol{C}\sum_{m=0}^{T}\boldsymbol{A}^m\boldsymbol{B}\boldsymbol{u}^T - \boldsymbol{C}\sum_{k=0}^{T-1}\sum_{m=T-k}^{T}\boldsymbol{A}^m\boldsymbol{B}\boldsymbol{u}^k \quad (7)\end{aligned}$$

In the above equation, omitting all but the second and third terms provides an invariant with no reliance on internal states. Omitted terms do not affect the invariant significantly and thus can be construed to constitute a tolerance term. Such observations lead to the following relation between the accumulated inputs and the outputs:

$$\sum_{t=0}^{T}\boldsymbol{y}^t = (\boldsymbol{C}\sum_{m=0}^{T}\boldsymbol{A}^m\boldsymbol{B} + \boldsymbol{D})\sum_{t=0}^{T}\boldsymbol{u}^t + \boldsymbol{\tau} \quad (8)$$

where the tolerance, $\boldsymbol{\tau}$, can be given as:

$$\boldsymbol{\tau} = \boldsymbol{C}\sum_{t=0}^{T}\boldsymbol{A}^t\left(\boldsymbol{s}^0 - \boldsymbol{B}\boldsymbol{u}^T\right) - \boldsymbol{C}\sum_{k=0}^{T-1}\sum_{m=T-k}^{T}\boldsymbol{A}^m\boldsymbol{B}\boldsymbol{u}^k \quad (9)$$

The term, $\sum_{t=0}^{T}\boldsymbol{A}^t$, converges in the limit to $(\boldsymbol{I}-\boldsymbol{A})^{-1}$ for large T. Convergence of the summation necessitates that $|\boldsymbol{A}| < 1$, which poses no extra constraint as it is in any case a required condition for the stability of linear digital systems. By utilizing this result, equation 8 can be simplified to produce:

$$\sum_{t=0}^{T}\boldsymbol{y}^t = \boldsymbol{\mathcal{I}}\sum_{t=0}^{T}\boldsymbol{u}^t + \boldsymbol{\tau} \quad (10)$$

where $\boldsymbol{\mathcal{I}}$ denotes

$$\boldsymbol{\mathcal{I}} = \boldsymbol{C}(\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{B} + \boldsymbol{D} \quad (11)$$

Equation 10 provides a simple relation between the inputs and outputs within a tolerance, $\boldsymbol{\tau}$, that can be utilized for concurrent error detection. Letting $\boldsymbol{\tau_{max}}$ denote the maximum magnitude of the tolerance, the following condition[1] provides a simple error checking mechanism.

$$\left\|\sum_{t=0}^{T}\left(\boldsymbol{y}^t - \boldsymbol{\mathcal{I}}\boldsymbol{u}^t\right)\right\| \leq \boldsymbol{\tau_{max}} \quad (12)$$

[1] We use the notation $||A||$ to denote a matrix whose entries are made up of the absolute value of the corresponding entries in the A matrix. The same notation is used for column vectors as in equation 12.

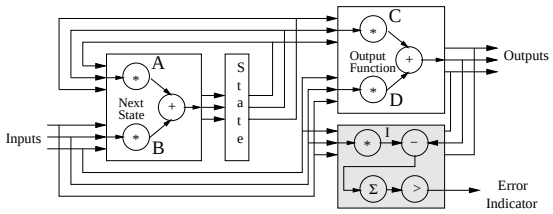

Figure 1. Online Fault Detection Hardware

Implementation of the error detection hardware based on equation 12 is shown shaded in figure 1. The concurrent error detection scheme for a system with one input and one output necessitates only a multiplier, a subtractor, an adder, a register, and a comparator.

3.2. Upper Bound on Tolerance Magnitude

Linearity of equation 9 guarantees that the upper and lower bound of the tolerance is equal in magnitude if the input range to the system is symmetrical about the origin. In linear systems, the dynamic range is usually chosen so that the negative and positive ranges are equal. This observation simplifies the consequent analysis of tolerance bounds by necessitating derivation only of the maximum value of the tolerance. The first term of equation 9 can easily be maximized for large T:

$$\|\boldsymbol{C}(\boldsymbol{I}-\boldsymbol{A})^{-1}\|\boldsymbol{s_{max}} + \|\boldsymbol{C}(\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{B}\|\boldsymbol{u_{max}} \tag{13}$$

An upper bound on the second part of the equation can be found by applying the triangle inequality after changing the order in which summations are performed. The reliance on T can be obviated in the last step of the following equation by utilizing the identity $\sum_{m=1}^{\infty} m\|\boldsymbol{A}\|^m = \|\boldsymbol{A}\|(\boldsymbol{I}-\|\boldsymbol{A}\|)^{-2}$.

$$\begin{aligned}
\boldsymbol{C}\sum_{k=0}^{T-1}\sum_{m=T-k}^{T}\boldsymbol{A}^m\boldsymbol{B}\boldsymbol{u}^k &\leq \sum_{m=1}^{T}\|\boldsymbol{C}\boldsymbol{A}^m\boldsymbol{B}\|\sum_{k=T-m}^{T-1}\left\|\boldsymbol{u}^k\right\| \\
&\leq \sum_{m=1}^{T} m\,\|\boldsymbol{C}\|\,\|\boldsymbol{A}\|^m\,\|\boldsymbol{B}\|\,\boldsymbol{u_{max}} \\
&\leq \|\boldsymbol{C}\|\|\boldsymbol{A}\|(\boldsymbol{I}-\|\boldsymbol{A}\|)^{-2}\|\boldsymbol{B}\|\boldsymbol{u_{max}}
\end{aligned} \tag{14}$$

While equation 14 provides a *constant* upper bound on the second term of the tolerance equation, it is quite pessimistic due to repetitive applications of the triangle inequality during its derivation. While the constant upper bound provided by equation 14 is not tight, it nonetheless is useful in guaranteeing that the second term of the tolerance in equation 9 is also bounded. Consequently, the overall tolerance, $\boldsymbol{\tau}$, can be bounded by:

$$\begin{aligned}
\boldsymbol{\tau} \leq \boldsymbol{\tau_{max}} &= \left\|\boldsymbol{C}(\boldsymbol{I}-\boldsymbol{A})^{-1}\right\|\boldsymbol{s_{max}} + \left\|\boldsymbol{C}(\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{B}\right\|\boldsymbol{u_{max}} + \\
&\quad \|\boldsymbol{C}\|\|\boldsymbol{A}\|(\boldsymbol{I}-\|\boldsymbol{A}\|)^{-2}\|\boldsymbol{B}\|\boldsymbol{u_{max}}
\end{aligned} \tag{15}$$

A tight but *time-dependent* upper bound can be attained by directly maximizing the second term of the tolerance equation 9. The result of the maximization is given in equation 16.

$$\sum_{k=0}^{T-1}\left\|\boldsymbol{C}\sum_{m=T-k}^{T}\boldsymbol{A}^m\boldsymbol{B}\right\|\boldsymbol{u_{max}} \tag{16}$$

For the elliptic filter coefficients in table 1, equation 15 determines the constant overall upper bound to be $[11.10]$. In case equation 16 is utilized for the upper bound of the second term, however, the overall bound is determined to be only $[3.63]$ at $T = 1,000,000$.

4. Fault Detection Analysis

In an accumulation-based concurrent error detection scheme, the fault detection capabilities of the system depend on the average behavior of the accumulated fault effects on the output. For average fault behavior analysis, we utilize the superposition principle. Independent of the fault model, the effect of a fault in a linear system can be modeled as an external input to the system at an appropriate point. A fault in the matrix vector multiplication for the next state function, either in the $\boldsymbol{As}^t$ or the $\boldsymbol{Bu}^t$ term, can be modeled as an additive external input, $\boldsymbol{f_s}$, a $k \times 1$ column vector, to the system. A fault in the matrix vector multiplication of the output function can also be modeled as an external input, $\boldsymbol{f_o}$, a $k \times 1$ column vector, to the system.

The modified equations for the system considering the external inputs $\boldsymbol{f_s}$ and $\boldsymbol{f_o}$ become:

$$\boldsymbol{s}^{t+1} = \boldsymbol{A}\boldsymbol{s}^t + \boldsymbol{B}\boldsymbol{u}^t + \boldsymbol{f_s}^t \tag{17}$$

$$\boldsymbol{y_{total}}^t = \boldsymbol{C}\boldsymbol{s}^t + \boldsymbol{D}\boldsymbol{u}^t + \boldsymbol{f_o}^t \tag{18}$$

where $\boldsymbol{y_{total}}^t = \boldsymbol{y}^t + \boldsymbol{y_f}^t$. By utilizing the superposition principle, the incremental fault effect at the output, $\boldsymbol{y_f}^t$, can be determined by setting the system inputs and initial states to zero. In order to derive the average effect on the output, we follow the steps in section 3.1 and arrive at:

$$\sum_{t=0}^{T}\boldsymbol{y_f}^t = \boldsymbol{C}\sum_{t=1}^{T}\sum_{k=0}^{t-1}\boldsymbol{A}^{t-1-k}\boldsymbol{f_s}^k + \sum_{t=0}^{T}\boldsymbol{f_o}^t \tag{19}$$

By taking expectations of both sides of the previous equation, the following result can be derived:

$$\begin{aligned}
\mathcal{E}\left(\sum_{n=0}^{T}\boldsymbol{y_f}^n\right) &= \boldsymbol{C}\sum_{n=1}^{T}\sum_{k=0}^{n-1}\boldsymbol{A}^k\mathcal{E}(\boldsymbol{f_s}) + T\mathcal{E}(\boldsymbol{f_o}) \\
&= T\boldsymbol{C}(\boldsymbol{I}-\boldsymbol{A})^{-1}\mathcal{E}(\boldsymbol{f_s}) - \\
&\quad \boldsymbol{C}\boldsymbol{A}(\boldsymbol{I}-\boldsymbol{A})^{-2}\mathcal{E}(\boldsymbol{f_s}) + T\mathcal{E}(\boldsymbol{f_o})
\end{aligned} \tag{20}$$

The linearity of the system, frequently invoked in the derivation through the superposition principle, ensures that

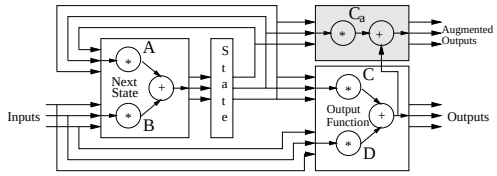

Figure 2. Augmented Output Function

average fault effects that propagate to component outputs are propagated to system outputs unaltered expect for being multiplied by a constant that depends on the system parameters, as can also be seen through equation 20.

A term by term analysis of equation 20 shows that the second term is constant and can thus be ignored for the purposes of fault detection analysis. The first term governs the faults in the next state function, while the third term governs the faults in the output function. We analyze faults on the next state function and output function separately.

The third term of equation 20 indicates that the fault effects accumulate over time unless the average value of the fault effects at the point of activation is zero. For faults with monotonic[2] effects, the average value will be non-zero. Furthermore, the effect of numerical inaccuracies on the fault detection scheme can be eliminated by utilizing unbiased rounding schemes[3]. Merging equations 12 and 20 through superposition results in the following sufficiency condition for the detection of faults in the output logic.

$$|T\mathcal{E}(\boldsymbol{f_o})| \quad > \quad \boldsymbol{\tau_{max}} \tag{21}$$

Equation 21 indicates that the detection latency of the faults in the output function is directly proportional to the tolerance and inversely proportional to the fault effects. Therefore, the higher the fault effect, the sooner will the faults be detected by the proposed scheme.

The first term of equation 20 indicates that detection of faults in the next state function depends on both the average value of the faults at the point of activation and the system parameters $\boldsymbol{C}$ and $\boldsymbol{A}$. The average value of the fault effects is multiplied by the *fault detection vector*, $\boldsymbol{C}(\boldsymbol{I}-\boldsymbol{A})^{-1}$. Detection of faults in the next state function requires that none of the terms of the fault detection vector be zero. The fault detection vector, depending on the system parameters, may not satisfy this criterion. In such cases, an additional output function may be provided to improve concurrent fault detection, as shown in equation 22. An implementation that incorporates this augmentation is given shaded in figure 2.

$$\boldsymbol{y_a^t} \quad = \quad \boldsymbol{y}^t + \boldsymbol{C_a s}^t \tag{22}$$

[2]Monotonic implementations of RT components for linear digital filters have been previously shown [1].

[3]Digital filter implementations with unbiased rounding schemes have been previously shown [1].

Invariant $\mathcal{I}$	Tolerance $\mathcal{T}_{max}$	Detection Vector $C(I-A)^{-1}$			
[0.9441]	[3.63]	[0.80	1.12	0.03	0.02]
[0.1778]	[2.21]	[0.19	-0.63	0.02	-0.03]
[0.1779]	[0.79]	[0.00	0.00	-0.08	0.34]

Table 4. System Parameters

The condition for fault detection, with the augmented output, becomes

$$(\boldsymbol{C}+\boldsymbol{C_a})(\boldsymbol{I}-\boldsymbol{A})^{-1} \neq 0 \tag{23}$$

It is desirable that a highly sparse $\boldsymbol{C_a}$ matrix be selected to reduce additional hardware overhead. While augmenting the output function for error detection increases the area overhead, utilizing terms from the multiplication of $\boldsymbol{As}^t$ in the next state function reduces the additional hardware requirement. Intermediate multiplication results from $\boldsymbol{As}^t$ can always be utilized as there are no restrictions on $\boldsymbol{C_a}$. Utilizing intermediate results from the next state function reduces the additional hardware requirement solely to adders by eliminating the need for the additional multipliers.

5. Fault Model and Experiments

We utilize monotonic implementations of the RT components which we previously outlined [1]; such implementations reflect the effect of the component internal stuck-at faults monotonically to the outputs, thus necessitating only examination of consequent input and output stuck-at faults. Additionally, faults in the multipliers for both next state and output functions can be modeled as if they affect directly the output of the adders, as also noted in section 4. The model we use consequently covers all stuck-at faults, unlike previous work [2] that relied on a somewhat functional bit flipping fault model.

Three elliptic filters are designed using Matlab®. The matrices for the state variable representation of the filters are given in tables 1, 2, and 3. Fault simulations using random patterns are performed on an RT level simulator developed in C. The invariant, the time varying tolerance at[4] $T = 1,000,000$, and the fault detection vectors for the three designs are all provided in table 4.

For the first elliptic filter, detailed fault coverage results are given in figure 3 by splitting the filter into 4 parts, corresponding to multiplication by $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, and $\boldsymbol{D}$. The figure indicates that the latency for the $\boldsymbol{A}$ and $\boldsymbol{B}$ pair and the $\boldsymbol{C}$ and $\boldsymbol{D}$ pair are very close, which is in agreement with the analysis outlined in section 4. The latency for the $\boldsymbol{A}$ and $\boldsymbol{B}$ pair exceeds that of the $\boldsymbol{C}$ and $\boldsymbol{D}$ pair as the two terms of the detection vector given in table 4 are quite small.

[4]$T = 1,000,000$ is selected as it exceeds comfortably consequent latencies.

$$\boldsymbol{A}=\begin{bmatrix} .419 & -.419 & .000 & .000 \\ .419 & .876 & .000 & .000 \\ -.015 & 1.845 & .750 & -.580 \\ -.005 & .612 & .580 & .808 \end{bmatrix} \boldsymbol{B}=\begin{bmatrix} .652 \\ .192 \\ .797 \\ .264 \end{bmatrix}$$

$$\boldsymbol{C}=\begin{bmatrix} -.003 & .403 & -.005 & .023 \end{bmatrix} \boldsymbol{D}=[.174]$$

Table 1. Elliptic Filter 1

$$\boldsymbol{A}=\begin{bmatrix} .832 & .471 & .000 & .000 \\ -.472 & .317 & .000 & .000 \\ -.374 & -.738 & .821 & .562 \\ -1.174 & -2.316 & -.562 & .765 \end{bmatrix} \boldsymbol{B}=\begin{bmatrix} .238 \\ .666 \\ .782 \\ 2.455 \end{bmatrix}$$

$$\boldsymbol{C}=\begin{bmatrix} -.293 & -.578 & -.015 & -.019 \end{bmatrix} \boldsymbol{D}=[.613]$$

Table 2. Elliptic Filter 2

$$\boldsymbol{A}=\begin{bmatrix} -.367 & -.259 & .633 & -.259 \\ .259 & -.106 & .259 & .894 \\ -.633 & .259 & .367 & .259 \\ -.259 & -.894 & -.259 & .106 \end{bmatrix} \boldsymbol{B}=\begin{bmatrix} .582 \\ .238 \\ -.582 \\ -.238 \end{bmatrix}$$

$$\boldsymbol{C}=\begin{bmatrix} .038 & .325 & .038 & .326 \end{bmatrix} \qquad \boldsymbol{D}=[.213]$$

Table 3. Elliptic Filter 3

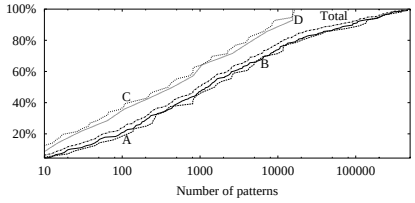

Figure 3. Fault coverage curves for elliptic filter 1

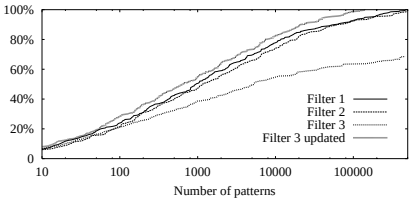

Figure 4. Fault coverage curves for all filters

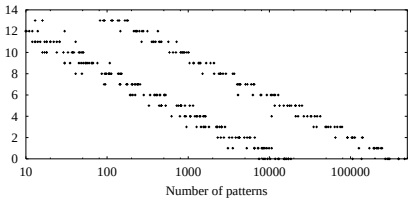

Figure 5. Bit Positions of Detected Faults

Fault coverage results for all three designs are provided in figure 4; they indicate that while the first two designs provide 100% coverage, the coverage level for the third design reaches only up to 60%. Two elements of the fault detection vector for the third design are zero, causing the fault effects not to be accumulated, thus resulting in coverage loss. Upon application of the augmented output scheme that we have proposed in section 4 with $\boldsymbol{C_a} = [0.1\ 0.1\ 0.0\ 0.0]$, we are able to obtain complete coverage levels of 100% with an additional overhead of two multipliers and two adders; this is shown as *Filter 3 updated* in the same figure 4. Complete fault coverages are attained around 500,000 patterns, with consequent latencies well below human response times at current IC frequencies.

Figure 5 provides an analysis of the fault detection sites in terms of their bit positions, with bit 0 indicating the least significant bit. The figure confirms that faults with large magnitude effects are detected earlier. The separation of the detection points into two regions is due to the fact that faults in the output function are detected earlier than faults in the next state function. While the errors at the output function are propagated to the output with no attenuation, the errors at the next state function are attenuated according to the value of the detection vector.

6. Conclusion

A concurrent error detection scheme for linear digital state variable systems is proposed. Through accumulation-based algorithmic approaches, the scheme detects faults concurrently by only observing system inputs and outputs. Unlike previously suggested schemes, the proposed fault detection scheme does not rely on access to internal states. Lack of access to internal states of the system introduces latency to fault detection, yet at the same time appreciably reduces hardware overhead incurred by such schemes.

Thorough mathematical analysis indicates that the proposed scheme is capable of providing concurrent error detection with minimal access to the system. The fault detection capability of the proposed scheme is also verified through experiments performed on three systems. The results of the experiments indicate that 100% fault detection is possible within a short time latency. The latency of detection is appreciably shorter for faults with larger magnitude effects.

The results attained in this work provide a basis for a low-cost fault tolerance scheme applicable to a wide range of DSP algorithms. Minimal access to design internals coupled with straightforward mathematical results for implementation makes the proposed scheme accessible to a large community of IC designers.

References

[1] I. Bayraktaroglu and A. Orailoglu. Cost effective digital filter design for concurrent test. In *ICASSP'2000*, pages 3323–3326, June 2000.

[2] A. Chatterjee and M. A. d'Abreu. The design of fault-tolerant linear digital state variable systems: theory and techniques. *IEEE Trans. on Computers*, 42(7):794–808, July 1993.

[3] K. H. Huang and J. A. Abraham. Algorithm-based fault tolerance for matrix operations. *IEEE Trans. on Computers*, 33(6):518–522, June 1984.

[4] J. Y. Jou and J. A. Abraham. Fault-tolerant FFT networks. *IEEE Trans. on Computers*, 37(5):548–561, May 1988.

[5] V. S. S. Nair and J. A. Abraham. Real-number codes for fault-tolerant matrix operations on processor arrays. *IEEE Trans. on Computers*, 39(4):426–435, April 1990.

[6] A. L. Narasimha Reddy and P. Banerjee. Algorithm-based fault detection for signal processing applications. *IEEE Trans. on Computers*, 39(10):1304–1308, October 1990.

[7] D. L. Tao and C. R. P. Hartmann. A novel concurrent error detection scheme for FFT networks. *IEEE Trans. on Parallel and Distributed Systems*, 4(2):198–221, February 1993.