WHICH CONCURRENT ERROR DETECTION SCHEME TO CHOOSE ?

Subhasish Mitra and Edward J. McCluskey
Center for Reliable Computing
Departments of Electrical Engineering and Computer Science
Stanford University, Stanford, California
http://crc.stanford.edu

Abstract

Concurrent error detection (CED) techniques (based on hardware duplication, parity codes, etc.) are widely used to enhance system dependability. All CED techniques introduce some form of redundancy. Redundant systems are subject to common-mode failures (CMFs). While most of the studies of CED techniques focus on area overhead, few analyze the CMF vulnerability of these techniques. In this paper, for the first time, we present simulation results to quantitatively compare various CED schemes based on their area overhead and the protection (data integrity) they provide against multiple failures and CMFs. Our results indicate that, for the simulated combinational logic circuits, although diverse duplex systems (with two different implementations of the same logic function) sometimes have marginally higher area overhead, they provide significant protection against multiple failures and CMFs compared to other CED techniques like parity prediction.

1. Introduction

Concurrent Error Detection (CED) techniques are widely used to enhance system dependability [Sellers 68, Kraft 81, Hsiao 81, Rao 89, Chen 92, Pradhan 96, Webb 97, Spainhower 99]. Almost all CED techniques function according to the following principle: Let us suppose that the system under consideration realizes a function f and produces output $f(i)$ in response to an input sequence i. A CED scheme generally contains another unit which independently predicts some special characteristic of the system-output $f(i)$ for every input sequence i. Finally, a *checker* unit checks whether the special characteristic of the output actually produced by the system in response to input sequence i is the same as the one predicted and produces an *error* signal when a mismatch occurs. Some examples of the characteristics of $f(i)$ are: $f(i)$ itself, its parity, 1's count, 0's count, transition count, etc. The architecture of a general CED scheme is shown in Fig. 1.1. Any CED scheme is characterized by the class of failures in the presence of which the system data integrity is preserved. By *data integrity*, we mean that the system either produces correct outputs or indicates erroneous situations when incorrect outputs are produced. In the literature on fault-tolerance, this property has been referred to as the *fault-secure* property [Siewiorek 92].

It may be noted that the general architecture of a CED scheme such as Fig.1.1 relies on the use of *hardware redundancy* (predictor and checker circuits) [Pradhan 96] for error-detection. Time redundancy techniques like *alternate-data-retry* and *recomputation with shifted operands* [Shedletsky 78, Patel 82] can also be used for concurrent error detection. Time redundancy directly affects the system performance although the hardware cost is generally less than that of hardware redundancy. The focus of this paper is on CED techniques using hardware redundancy.

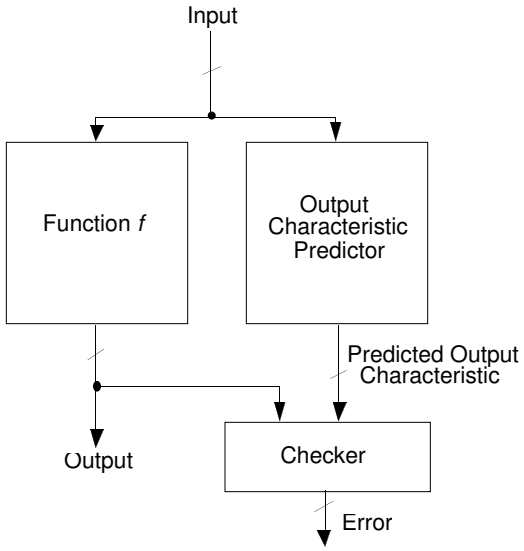

Figure 1.1. General architecture of a concurrent error detection scheme

Several CED schemes have been proposed and used commercially for designing reliable computing systems [Hsiao81, Chen 92, Webb 97, Spainhower 99]. These techniques mainly differ in their error-detection capabilities and the constraints they impose on the system design. There are many publications on system design with concurrent error detection. These include designs of datapath circuits (like adders, multipliers, etc.) [Sellers 68, Nicolaidis 93, Nicolaidis 97], and general combinational and sequential logic circuits [Aksenova 75, Jha 93, De 94, Touba 97, Zeng 99] with concurrent error detection. Checker circuit designs for concurrent error detection are described in [Wakerly 78, McCluskey 90]. Almost all publications on CED focus on their area/performance overhead. Reliability analysis of systems with concurrent error detection is presented in [Ramamoorthy 75]. However, the systems considered are restricted to those with redundancy through replication.

All the above-mentioned CED techniques guarantee system data integrity against single faults. However, these

CED schemes are vulnerable to multiple faults and common-mode failures. Common-mode failures are a special and very important cause of multiple faults. *Common-mode failures* (CMFs) produce multiple faults, occurring generally due to a single cause; the system data integrity is not guaranteed in the presence of CMFs. These include design mistakes and operational failures that may be due to external (such as EMI, power-supply disturbances and radiation) or internal causes [Avizienis 84, Lala 94]. CMFs in redundant VLSI systems are surveyed in [Mitra 00a]. *Design diversity* has been proposed in the past to protect redundant systems against common-mode failures. While most of the previous efforts towards definition of design diversity were qualitative, in an earlier paper [Mitra 99a] we developed a metric to quantify diversity among several designs and used this metric to analyze the reliability of redundant systems in the presence of CMFs.

It may be argued that, unlike systems with duplication, concurrent error detection techniques based on error detecting codes (e.g. parity, etc.) introduce inherent diversity in the system. Thus, qualitatively, these systems must be well-protected against CMFs.

The problem studied in this paper is to compare five CED techniques for general combinational logic circuits based on their area overhead and their vulnerability to multiple failures and CMFs. The CED techniques considered are those based on identical and diverse duplication, parity prediction and Berger and Bose-Lin codes. These techniques are general and can be used for any system, unlike some other application-specific error detection techniques such as [Mahmood 84, Jou 88, Huang 00].

This paper is organized as follows. Section 2 presents a brief overview of various CED techniques. In Sec. 3, we present simulation results to compare these CED techniques. Section 4 describes analysis techniques to quantify the vulnerability of various CED schemes to multiple failures and CMFs. Some attempts to explain the simulation results of Sec. 3 and some open questions are reported in Sec. 5. The use of transition counting and residue codes for concurrent error detection is discussed in Sec. 6. Section 7 presents a system-level view of the CED techniques studied in this paper. Finally, we conclude in Sec. 8.

2. An Overview of Various CED Techniques

2.1. Duplex System

A duplex system is an example of a classical redundancy scheme that can be used for concurrent error detection [Sellers 68, Kraft 81, Sedmak 78]. Figure 2.1 shows the basic structure of a duplex system. Duplication has been used for concurrent error detection in numerous systems including the Bell Switching System [Kraft 81], systems from companies like Stratus and Sequoia [Pradhan 96] and also in the IBM G5 and G6 processors [Webb 97, Spainhower 99].

In any duplex system there are two modules (shown in Fig. 2.1 as Module 1 and Module 2) that implement the same logic function. The two implementations are not necessarily the same. A comparator is used to check whether the outputs from the two modules agree. If the outputs disagree, the system indicates an error. For a duplex system, data integrity is preserved as long as both modules do not produce identical errors (assuming that the comparator is fault-free). Since the comparator is crucial to the correct operation of the duplex system, special self-checking comparator designs (e.g., two-rail checker [McCluskey 90]) that guarantee data integrity against single comparator faults must be used.

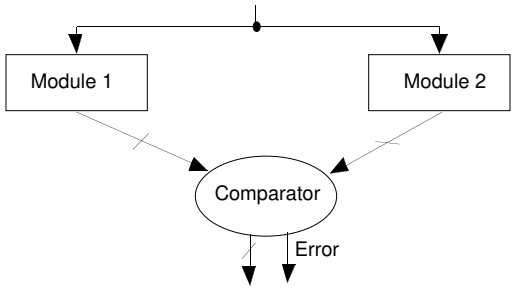

Figure 2.1. A Duplex System

2.2. Parity Prediction

Parity prediction is a widely used CED technique. The even/odd parity function indicates whether the number of 1's in a set of binary digits is even or odd. Techniques for designing datapath logic circuits and general combinational circuits with parity prediction have been described in [Sellers 68, Kraft 81, Nicolaidis 93, Nicolaidis 97, De 94, Touba 97]. CED techniques with parity prediction in sequential circuits are described in [Zeng 99]. Figure 2.2 shows the basic architecture of a system with concurrent error detection using a single parity bit. The circuit has *m* outputs and is designed in such a way that there is no sharing among the logic cones generating each of the outputs. Thus, a single fault can affect at most one output bit position. The parity of the outputs is predicted independently. The parity checker checks whether the actual parity of the outputs matches the predicted parity [McCluskey 90].

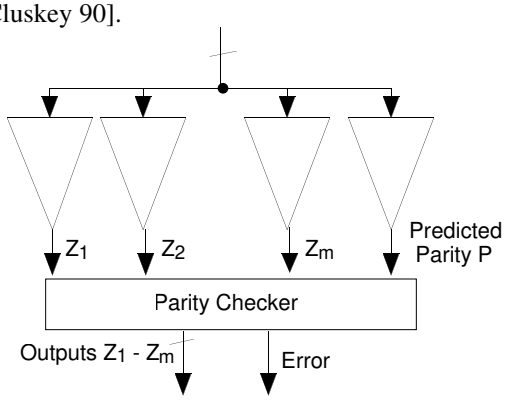

Figure 2.2. Parity prediction using a single parity bit

The restriction of no logic sharing among different logic cones can result in large area overhead for circuits with a single parity bit. Hence, the idea of using a single

parity bit has been extended to multiple parity bits. This technique partitions the primary outputs into different parity groups. Sharing is allowed only among logic cones of the outputs that belong to different parity groups. There is a parity bit associated with the outputs in each parity group. The outputs of each parity group are checked using a parity checker. Figure 2.3 shows the general structure of a combinational logic circuit with two parity groups.

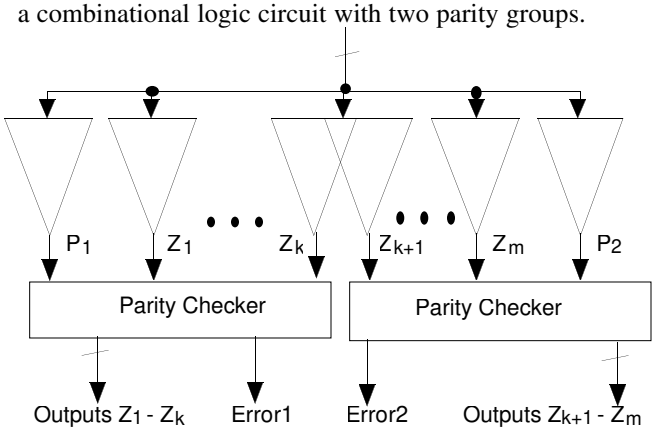

Figure 2.3. Multiple parity bits for concurrent error detection

In the circuit of Fig. 2.3, there are two parity groups G_1 and G_2. The parity group G_1 contains the outputs Z_1, …, Z_k. P_1 is the predicted parity for this parity group. It predicts the parity of the primary outputs in G_1. The parity group G_2 contains the outputs Z_{k+1}, …, Z_m. P_2 is the predicted parity bit associated with this parity group. There is sharing between logic cones corresponding to the outputs Z_k and Z_{k+1}. No logic sharing is allowed among the cones corresponding to outputs Z_1, …, Z_k (Z_{k+1}, …, Z_m). Sharing is allowed among logic cones corresponding to other output groups such as Z_h and Z_j, $1 \leq h \leq k$, $k+1 \leq j \leq m$.

2.3. Unidirectional Error Detecting Codes

CED techniques based on unidirectional error detecting codes have been proposed in the past. A unidirectional error detecting code assumes that all errors are unidirectional; i.e., they change 0s to 1s or 1s to 0s but never both at the same time. Two unidirectional error detecting codes used for concurrent error detection are Berger codes [Berger 61], and Bose-Lin codes [Bose 85].

For the Berger code, a code-word is formed by appending a binary string representing the number of 0s (or the bit-wise complement of the number of 1s) in the given information word. Thus, for an information word consisting of n bits, the Berger code requires $\lceil log_2 n \rceil$ extra bits to represent the number of 0s (or the bit-wise complement of number of 1s) in the information word. The Berger code has the capability of detecting all unidirectional errors. Figure 2.4 shows a concurrent error detection technique using Berger codes.

Since the Berger code is a unidirectional error detection code, it is important to ensure that a single fault causes unidirectional errors at the outputs. This imposes a restriction that the logic circuits should be synthesized in such a way that they are inverter-free [Jha 93]. Inverters can only appear at the primary inputs. In general, for Berger codes used to detect unidirectional errors on communication channels, the check-bits represent the bit-wise complement of the number of 1's in the information word. "However, since concurrent error detection techniques are designed to guarantee data integrity in the presence of single faults, a single fault can affect either the actual logic function or the logic circuit that predicts the number of 1's at the output but never both at the same time (since there is no logic sharing between the actual circuit and the circuit that predicts the number of 1's)". Thus, we need not obtain a bit-wise complementation of the number of 1's [Das 98]. The checker design for Berger codes is described in [Marouf 78].

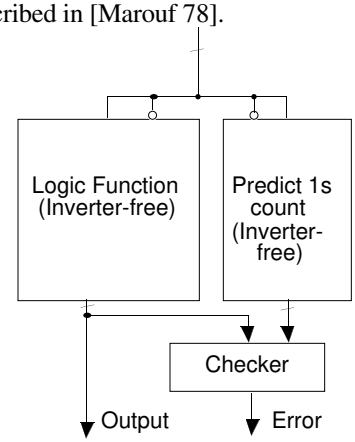

Figure 2.4. Concurrent Error Detection Using Berger Codes

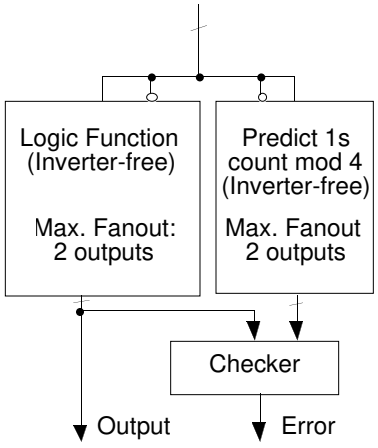

Figure 2.5. Concurrent Error detection using Bose-Lin codes

Bose-Lin codes are capable of detecting t-bit unidirectional errors in the code-word. The constructions of Bose-Lin codes for $t = 2$ and $t = 3$ are given in [Bose 85]. Design of logic circuits with concurrent error detection based on Bose-Lin codes has been reported in [Das 98]. Figure 2.5 shows the architecture of a system with concurrent error detection based on 2-bit unidirectional error

detecting Bose-Lin code. Just like Berger codes, we want the circuit to be inverter-free (except at the primary inputs) so that any single fault creates unidirectional errors at the outputs. We also need a restriction on the amount of logic sharing since the code is capable of detecting at most 2 unidirectional errors. The restriction is that, any logic gate in the circuit can be shared by the logic cones of at most two primary outputs. Checker circuits for Bose-Lin codes can be obtained from [Jha 91].

3. Simulation Results

In this section, we provide simulation results to compare the five CED schemes (identical and diverse duplication, parity prediction, Berger codes and Bose-Lin codes), described in Sec. 2, based on their area overhead and their vulnerability of different CED schemes to multiple failures and CMFs. The simulation results show the superiority of diverse duplication over other conventional CED schemes for the simulated designs.

We considered some combinational logic circuits from the MCNC 91 benchmark suite for simulation purposes. We used the *Sis* tool [Sentovich 92] for synthesizing circuits. For designing a diverse duplex system (with different implementations), we generated truth tables with complemented outputs and synthesized them using *Sis*. Finally, we added inverters at the outputs of the resulting implementation. For duplex systems, all the synthesis optimizations can be applied. We used *espresso* for two-level minimization and *script.rugged* available with the *Sis* tool for multi-level optimization and mapped the circuits to the LSI Logic G10p technology library [LSI 96]. For the CED scheme with parity prediction we used the technique in [Touba 97]. For synthesizing circuits with Berger codes, we must ensure that the individual circuits are inverter-free. The synthesis technique has been described in [Jha 93]. We used algebraic transformations (using *script.algebraic* available with *Sis*) during multi-level logic synthesis so that the circuits are inverter-free. For synthesizing circuits with Bose-Lin codes, a similar approach was used. However, we have to limit the fanout structure such that a gate can be shared by a maximum of two output functions. The technique in [Das 98] was used for synthesizing logic circuits with Bose-Lin codes.

Table 3.1. Area overhead of various CED schemes

Circuit	Identical Duplex	Diverse Duplex	Parity	Berger Code	Bose - Lin
Z5xp1	**822**	836	840	1335	1068
inc	743	751	**692**	854	807
squar5	507	485	**465**	627	570
ex5.20	646	649	**593**	815	755
misex1	412	423	**367**	468	488
sao2	754	787	**748**	983	864
rd73	**474**	480	683	853	763
rd84	**642**	684	971	1135	1056

Table 3.1 shows a comparison of the area overhead (in terms of the G10p cell areas reported by the *Sis* tool) of various CED schemes for eight MCNC benchmark circuits. It is clear from Table 3.1 that the area overhead of CED techniques based on Berger codes and Bose-Lin codes are much higher than those based on parity prediction or duplication. For many circuits, the area overhead of parity prediction is marginally less than that of duplication. Similar observations have been made in [Zeng 99]. Hence, for the rest of this paper we focus mainly on CED techniques based on duplication and parity prediction. Next, we present simulation results on the vulnerability of CED techniques to multiple failures and CMFs (permanent or temporary).

First, we consider the case of permanent faults. In dependable systems, it is realistic to assume a corrective action is initiated after the system generates an error signal. Thus, for any system with concurrent error detection, data integrity is guaranteed as long as the system does not produce an undetected corrupt output before indicating the presence of an error. In the following discussion, we focus on systems consisting of combinational logic circuits. However, the entire discussion can be extended for sequential logic circuits.

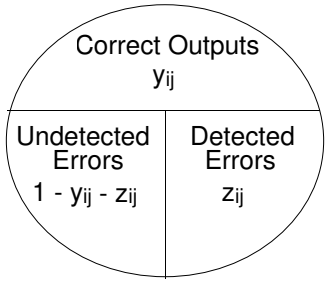

Figure 3.7. Various components of output events of a system with CED along with their probabilities

The probability that the data integrity of a combinational logic system is guaranteed up to time t in the presence of a fault pair (f_i, f_j) is derived in the following way. *Given an input distribution, let us suppose that the probability that the system produces correct outputs in the presence of (f_i, f_j) is $y_{i,j}$; the probability that the system produces incorrect outputs that can be detected is $z_{i,j}$.* Figure 3.7 shows a Venn diagram to explain $y_{i,j}$ and $z_{i,j}$.

Assuming that the fault pair is permanent, the probability that the system data integrity is guaranteed up to time t (after the occurrence of the fault pair) is:

$$y_{i,j}^t + \sum_{k=1}^{t} y_{i,j}^{k-1} z_{i,j} = y_{i,j}^t + \frac{z_{i,j}}{1 - y_{i,j}} (1 - y_{i,j}^t)$$

The above expression can be derived from the fact that the system must either produce correct outputs up to time t or indicate an error signal for the first time without producing any corrupt data before t.

From the above expression for data integrity, it is clear that the term $w_{i,j} = \dfrac{z_{i.j}}{1 - y_{i,j}}$ plays an important role in determining the system data integrity up to time t. This term $w_{i,j}$, *the detected fraction*, is the fraction of output

error events detected in the presence of the fault pair (f_i, f_j). If the value of this term is 1 the system either produces correct outputs or indicates erroneous situations when incorrect outputs are produced. If the value is 0 the system never produces any error signal when incorrect outputs are produced. Note that, if a CED-based system produces correct outputs for all input combinations even in the presence of a fault, then the fault is redundant.

We used the following procedure to estimate the protection against multiple and common-mode failures provided by CED techniques based on duplication and parity prediction. For each single-stuck-at fault f_i in each of these circuits, we simulated exhaustively all fault pairs and input combinations to identify another single-stuck-at fault f_j in the same circuit that had the minimum value of $w_{i,j}$. Hence, the fault pair (f_i, f_j) can be regarded as a *worst-case permanent fault pair*. Finally, we averaged the $w_{i,j}$'s over all the worst-case permanent fault pairs to obtain the average value of the worst-case detected fraction of incorrect outputs. Such a metric is pessimistic because we are considering the worst-case permanent fault pairs. The results are shown in Table 3.2. The benchmark circuits are small enough so that exhaustive simulation is possible.

Table 3.2. Average value of the detected fraction of incorrect outputs for the worst-case permanent faults

Circuit	Identical Duplex	Diverse Duplex	Parity
Z5xp1	0	**0.70**	0.46
inc	0	**0.68**	0.45
squar5	0	**0.55**	0.53
ex5.20	0	**0.30**	0.20
misex1	0	**0.54**	0.40
sao2	0	**0.60**	0.06
rd73	0	**0.60**	0.40
rd84	0	**0.66**	0.51

Table 3.2 demonstrates the advantages of using diverse duplex systems over other CED schemes. It may be noted that for diverse duplex systems, we found several worst-case permanent fault pairs with the value of $w_{i,j}$ equal to 1. This means that, even in the worst-case, system data integrity is guaranteed for these fault pairs in the diverse duplex system. However, we did not find any such worst-case permanent fault pairs for systems with parity prediction.

In addition to the above worst-case analysis, we studied the CED techniques in various other ways. These simulations also indicate that diverse duplication provides better data integrity compared to identical duplication or parity prediction against multiple and common-mode failures. In Tables 3.3a (3.3b), for the above benchmark circuits, we report the percentage of all worst-case permanent fault pairs in parity prediction (diverse duplication) with greater or equal detectability of incorrect outputs compared to the worst-case permanent fault pairs in diverse duplication (parity prediction). For example, for the Z5xp1 circuit, 80% of the worst-case permanent fault pairs in diverse duplication have values of incorrect output detectability greater than or equal to those in parity prediction (Table 3.3b).

Table 3.3. (a) Percentage of worst-case permanent fault pairs in parity prediction with greater (or equal) incorrect output detectability compared to those in diverse duplication. (b) Percentage of worst-case permanent fault pairs in diverse duplication with greater (or equal) incorrect output detectability compared to those in parity prediction

(a)

Circuit	Percentage
Z5xp1	20 %
inc	22 %
squar5	49 %
ex5.20	50 %
misex1	35 %
sao2	10 %
rd73	25 %
rd84	23 %

(b)

Circuit	Percentage
Z5xp1	80 %
inc	81 %
squar5	55 %
ex5.20	60 %
misex1	70 %
sao2	90 %
rd73	77 %
rd77	77 %

It may be argued that CMFs and multiple failures may have temporary effects and it may be inaccurate to model them as permanent faults. Next, we present simulation results to compare the vulnerability of various CED schemes to temporary CMFs and multiple failures (possibly due to transient failures like radiation upsets, power-supply disturbances, etc. or intermittent failures) that persist for a single clock cycle. The vulnerability of a CED scheme to a fault pair (f_i, f_j) resulting from such a failure is given by $d_{i,j}$ *which is the conditional probability that the system either produces correct outputs or generates an error signal if an incorrect output is produced in the presence of* (f_i, f_j) *for a given input distribution.* Note that, for duplex systems, $d_{i,j}$ is the same as the diversity with respect to the fault pair (f_i, f_j) as described in [Mitra 99a].

Table 3.4. Average value $d_{i,j}$'s of the worst-case temporary fault pairs

Circuit	Diverse Duplex	Parity Prediction
Z5xp1	**0.90**	0.70
inc	**0.92**	0.78
squar5	**0.90**	0.86
ex5.20	**0.89**	0.68
misex1	**0.90**	0.70
sao2	**0.93**	0.57
rd73	**0.90**	0.80
rd84	**0.88**	0.70

For simulation purposes, in each of these benchmark circuits with CED, for each single-stuck-at fault f_i, we simulated exhaustively all fault pairs and input combinations to identify another single-stuck-at fault f_j in the same circuit that had the minimum value of $d_{i,j}$. Hence, the fault pair (f_i, f_j) can be regarded as a *worst-case temporary fault pair*. Finally, we averaged the $d_{i,j}$'s over all the worst-case temporary fault pairs. These numbers are

reported in Table 3.4 for CED schemes based on diverse duplication and parity prediction. The benchmark circuits are small enough so that exhaustive simulation is possible.

The simulation results in this section demonstrate the advantages of diverse duplication in providing protection against multiple failures and CMFs compared to other CED schemes. However, the major problem with diverse duplication is to develop techniques for synthesizing logic functions with diversity. We have investigated some techniques for designing two implementations of any given combinational logic circuit in order to maximize diversity [Mitra 00b].

4. Analysis of Vulnerability to Multiple Failures and CMFs

The vulnerability of a duplex system to multiple failures and CMFs can be quantified using the idea of the design diversity metric presented in [Mitra 99a, Mitra 99b] and not repeated here.

For systems with parity prediction, if a single fault or multiple faults affect a single logic cone, the data integrity of the system is preserved. However, if a failure causes faults in two logic cones in the same parity group, the data integrity is not guaranteed.

A CED technique with a single parity bit is one with a single parity group. Referring to Fig. 2.3, let us suppose that fault f_i affects the parity prediction logic of P_2 and the fault f_j affects the part of the logic that is shared by primary outputs Z_k and Z_{k+1}. Let V_i be the *set of input combinations in response to which the parity prediction logic produces an incorrect output in the presence of* f_i. Similarly, let V_j be the *set of input combinations in response to which the cone of logic affected by* f_j *produces an incorrect output for* Z_{k+1} *but not for* Z_k. If the fault f_j produces errors on output Z_k, the error will be detected by the parity bit P_1 and data integrity will be preserved. The value of $d_{i,j}$ is $1 - \frac{|V_i \cap V_j|}{2^n}$, where n is the number of inputs of the logic circuit in Fig. 2.3.

The system considered in Table 4.1 has four outputs and two parity groups. The first parity bit is the parity of the first two outputs and the second parity bit is the parity of the remaining two outputs. Also assume that there is sharing between the logic cones of the first and the third output function. The fault-free outputs and the corresponding parity bits are shown in Table 4.1. Let us suppose that a CMF manifests itself as a single-stuck-at fault pair (f_i, f_j), where f_i affects the logic shared by the first and third primary outputs and f_j affects the logic cone that predicts the first parity bit. The faulty outputs are shown in the last two columns of Table 4.1. For the first and the fourth input combinations, an error will be reported by the parity checker corresponding to the second parity bit. For the second input combination, an error will be reported by the checker corresponding to the first parity bit. For the third input combination, the system produces erroneous outputs and none of the checkers can detect this erroneous situation. Thus, the data integrity is compromised for the third input combination only.

Table 4.1. Diversity calculation for the fault pair (f_i, f_j) in a CED scheme with two parity bits

Inputs	Fault-free outputs	Fault-free parity	Faulty outputs	Faulty parity
00	0 1 0 0	0 1	**1** 1 **1** 0	**1** 1
01	1 0 0 1	0 0	1 0 0 1	**1** 0
10	0 0 1 1	1 1	**1** 0 1 1	**0** 1
11	1 1 1 1	1 1	1 1 **0** 1	**0** 1

Note that, for any fault pair (f_i, f_j), $d_{i,j} = y_{i,j} + z_{i,j}$. Hence, $y_{i,j}$ and $z_{i,j}$ can also be calculated for the fault pair (f_i, f_j) using techniques similar to those used for calculating $d_{i,j}$.

5. Theoretical Analysis and Open Questions

In this section, we present some attempts to provide a theoretical explanation of the simulation results reported in Sec. 3. It is clear from our discussions in Sec. 4 that the analysis of vulnerability of various CED schemes to multiple failures and CMFs is dependent on the $d_{i,j}$ and $z_{i,j}$ values of different faults. Given these values, the analysis is simple. However, it may be very difficult to deduce relationships among the sets of $d_{i,j}$ and $z_{i,j}$ values of faults in a system with hardware duplication and a system with parity checking. This is because the constraints used to synthesize the systems with different CED techniques are different. For example, CED techniques based on a single parity bit do not allow any logic sharing among the logic cones corresponding to different outputs. On the other hand, CED techniques using hardware duplication do not impose any fanout or logic sharing restrictions within a module. It has been demonstrated in [Mitra 99b, Mitra 00b] that fanout restrictions and logic sharing affect the detectability and the $d_{i,j}$ values of fault pairs in redundant systems. In this section, we analyze these systems based on simplistic error models (e.g., the Bernoulli or the q-ary error model used for signature analysis [Saxena 97]) to provide an insight into the simulation results. Note that, these error models have many drawbacks and hence, the simplistic assumptions associated with these models are questionable [Saxena 97].

For the Bernoulli error model with parameter p, it is assumed that the probability that a fault produces an error on any output bit is p and is independent of errors on other outputs. Thus, the probability that a particular error vector (obtained by XOR-ing the fault-free and the faulty output) with i errors appears (for an n-output circuit) is $p^i(1-p)^{n-i}$. The probability that any arbitrary circuit produces erroneous outputs in the presence of a fault f_i is $\sum_{i=1}^{n} \binom{n}{i} p^i (1-p)^{n-i}$. Hence, for a system with diverse

duplication, the expected value of $(1 - d_{i,j})$ for (f_i, f_j) is $\sum_{i=1}^{n}\binom{n}{i}p^{2i}(1-p)^{2n-2i} = [p^2+(1-p)^2]^n - (1-p)^{2n}$. This is because the system data integrity is not preserved only when both modules produce identical errors.

For a system with parity prediction using a single parity bit, consider a fault pair (f_i, f_j) where f_i and f_j affect logic cones corresponding to outputs g and h, respectively. If the probability that f_i (f_j) produces an error in any output bit for a general logic circuit with no restrictions on logic sharing is p, then the probability that both f_i and f_j produce errors at outputs g and h, respectively, at the same time is p^2. Since there is no sharing among logic cones in a circuit with parity prediction, the probability that f_i (f_j) produces an error on output g (h) is p; however, the probability that f_i (f_j) produces an error on any other output is 0. Thus, the expected value of $(1 - d_{i,j})$ for a fault pair (f_i, f_j) is p^2. Note that, the detectability values of faults f_i and f_j have reduced drastically for the system with parity prediction; this will produce lower values of $(1 - d_{i,j})$ compared to diverse duplication for practical values of p (< 1). This is not true as shown by the simulation results. This is because the assumption of independence of errors on different output bits is not true for general logic circuits.

However, suppose that, for a diverse duplex system, we have one implementation which has no logic sharing among the different output cones and the other implementation does not have any constraint on the amount of logic sharing. Let us suppose that fault f_i affects the logic cone of output g in the first implementation and fault f_j affects the second implementation. The probability that f_i produces an error on output bit g is p. However, since there is no logic sharing among the different output cones, f_i does not affect the other output bits. Hence, the data integrity of the diverse duplex system is not preserved in the presence of (f_i, f_j) only when f_j produces an error on output g and no error on other output bits in the second implementation. The probability of this event is $p(1-p)^{n-1}$. Hence, the expected value of $(1 - d_{i,j})$ for a fault pair (f_i, f_j) is $p^2(1-p)^{n-1}$ which is less than p^2. Thus, in this scenario, even with the Bernoulli model we find that the data integrity of a diverse duplex system is better than that of parity prediction. For convenience of the above analysis, it is assumed that both faults f_i and f_j produce error at any output bit with probability p. However, similar analysis can be performed and similar conclusions can be reached when the value of parameter p is different for f_i and f_j.

On the other extreme, we can consider the q-ary model [Pradhan 91]. For the q-ary error model, it is assumed that, in an n-output circuit and for a fault f with detectability q (probability that the fault produces incorrect outputs), the probability of any non-zero error vector (obtained by xoring the fault-free and faulty responses) is $\frac{q}{2^n-1}$. Note that, there are $2^n - 1$ non-zero error vectors. Hence, for a system with diverse duplication, the value of $(1 - d_{i,j})$ for a fault pair (f_i, f_j) is $\frac{q^2}{2^n-1}$. Note that, for a system with identical duplication, the expected value of $(1 - d_{i,j})$ for a worst-case fault pair (f_i, f_j) is q (since, the worst-case fault pairs affect identical leads in both modules).

For a system with parity prediction using a single parity bit, consider a fault pair (f_i, f_j) where f_i and f_j affect logic cones corresponding to outputs g and h, respectively. If the detectability of f_i in a general logic circuit (with no restrictions on logic sharing) is q, then the detectability of f_i in the circuit with parity prediction is approximately $\frac{q}{2}$. This is because, out of $2^n - 1$ error vectors, 2^{n-1} produce an error on output bit g; hence, the probability that the fault f_i produces an error on output g in a general logic circuit is $\frac{q2^{n-1}}{2^n-1}$ which is approximately $\frac{q}{2}$. Hence, the expected value of $(1 - d_{i,j})$ for a worst-case fault pair (f_i, f_j) in a circuit with parity prediction using a single parity bit is $\frac{q^2}{4}$ (when both the faults produce errors) which is more than $\frac{q^2}{2^n-1}$. Similar analysis can be performed for circuits with multiple parity bits. Hence, diverse duplication provides better data integrity against multiple failures and CMFs compared to identical duplication and parity prediction. For convenience of the above analysis, it is assumed that both faults f_i and f_j have the same detectability q. However, similar analysis can be performed and similar conclusions can be reached when f_i and f_j have different detectability values.

Table 5.1. $(1 - d_{i,j})$ value for fault pair (f_i, f_j)

	Diverse Duplex	Parity
Bernoulli model	$[(1-p)^2+p^2]^n - (1-p)^{2n}$ * $p^2(1-p)^{n-1}$ **	p^2
q-ary model	$q^2(2^n-1)^{-1}$	$0.25q^2$

* - Both implementations have no fanout restrictions
** - One implementation has no output cone sharing

The results presented in this section are summarized in Table 5.1. Note that, the Bernoulli and the q-ary models may not be realistic for many logic circuits as pointed out in [Saxena 97]. Hence, we reiterate that the problem of developing more sophisticated and elegant models for theoretically analyzing the vulnerability of various CED techniques to multiple failures and CMFs is open.

6. Transition Count and Residue for CED

As mentioned in Sec. 1, any CED technique predicts a particular characteristic of the system output. The output characteristics considered in the previous sections are the output itself, parity functions and 1's (or 0's count). Some other possible output characteristics are transition count and residue modulo some number.

Transition counting has been used in the past as a compaction technique for circuit responses during off-line test [Hayes 76]. For a CED scheme based on transition count, the special output characteristic is the number of up ($0 \rightarrow 1$) and down ($1 \rightarrow 0$) transitions in a given output vector. Transition counting is not a favorable method for concurrent error detection. First, the maximum value of the total number of transitions (up and down) in an n-bit binary word is n-1. Thus, the number of bits needed to represent the number of transitions is equal to the number of bits required to represent the number of 1's (or 0's) in the same word. Transition counting has another serious problem. Suppose that the correct output word from a system is 100100. The number of up-transitions ($0 \rightarrow 1$) in the output word is 1 and the number of down-transitions ($1 \rightarrow 0$) is 2. Suppose that in the presence of a fault that causes a single error, the output word is changed to 100110. The number of up and down transitions in the erroneous output word is the same as that of the correct output word; hence, this error is not detected.

Residue codes are used for concurrent error detection in mainly datapath elements like adders, multipliers, etc. [Langdon 70, Avizienis 71]. Given an n-bit output vector, the output characteristic captured by a residue code *modulo* b is the binary representation of the number $x = y \bmod b$, where y is the n-bit number represented by the given output word. The recommended value of b is of the form $2^m - 1$. When $b = 3$, we need two bits to represent the residue of any number. For a CED scheme using residue checking modulo 3, there cannot be any logic sharing between any cones corresponding to any two primary outputs. If there is logic sharing between two cones corresponding to bit positions i and j ($i > j$), a single fault can cause errors in these two positions. Suppose that ($i - j$) is even and the correct output word has 0's in bit positions i and j. If a single fault causes the bit positions i and j to be flipped to 1, the resulting error will be divisible by 3 and will not be detected. Similar arguments can be made for the case with ($i - j$) is odd. Thus, we need two extra bits even though there cannot be any logic sharing unlike parity checking where we need only a single bit. Our synthesis results for $b = 3$ and 7 also show that the area required for a CED technique based on residue checking is very high for general logic circuits. Note that, this result is not true for datapath logic circuits like adders and multipliers because, simple general schemes can be devised to predict the residue of sum or product of two numbers (arithmetic coding). This result is also supported by the following observation in [Langdon 70]: "the residue mod 3 check adder is not economical unless the addition operands are already provided with the mod 3 check bits".

7. System-Level Issues

In the previous sections, we mainly focused on CED techniques for combinational logic blocks. In Fig. 7.1 we present a system-level view of concurrent error detection. The system in Fig. 7.1a contains a combinational logic block implementing a logic function f; the logic block obtains its inputs from register X and the outputs are stored in register Z.

In Fig. 7.1b, we present a duplication-based CED technique (identical or diverse) for the system in Fig. 7.1a. The combinational logic blocks $N_1(f)$ and $N_2(f)$ implement function f. Registers X and Z and the system bus are duplicated; this can possibly cause high area overhead. In order to create diversity in the register contents, register X_2 (Z_2) can store the complemented forms of the contents of register X_1 (Z_1). Figure 7.1c presents a CED scheme based on parity prediction for the system in Fig. 7.1a. Each register has a single parity bit (P_X for X and P_Z for Z). It has been demonstrated in Sec. 3, through simulation, that the area overhead of combinational logic blocks with parity prediction is marginally less than that of duplication; however, if the number of register flip-flops and bus lines are counted, the scheme in Fig. 7.1c has significantly less logic area overhead than Fig. 7.1b.

Figure 7.1d presents a CED scheme that uses diverse duplication for combinational logic blocks and parity prediction for registers and bus lines. Thus, we can achieve significant improvement in protection against multiple and common-mode failures (through diverse duplication) while the total area overhead is comparable to that of parity prediction (Fig. 7.1c). For this purpose, we need a tree of XOR gates, as shown in Fig. 7.1d. The CED scheme in Fig. 7.1d needs two extra 2-input XOR gates and one 2-input OR gate (XOR-tree and the equality checker) for each output of the combinational logic block compared to the CED scheme in Fig. 7.1c. Note that, the XOR tree may have significant delay overhead. This delay overhead can be reduced by increasing the number of parity bits (i.e., the number of extra flip-flops in the registers). Interesting problems analyzing this area-delay trade-off can be studied in this context. The XOR tree in Fig. 7.1d can be eliminated if the parity bit of the register is generated from a dual-rail checker used to check the outputs of the combinational logic [Nicolaidis 93]. Routing overhead of the designs in Fig. 7.1b, 7.1c and 7.1d has not been considered in the above discussion.

8. Conclusions

The theory and instrumentation of various concurrent error detection techniques have been subjects of active research interest since the late 1950s till today. However, no systematic study on the vulnerability of these CED schemes to multiple failures and CMFs has been reported in the past. In this paper, for the first time, we provided analytical formulas and simulation results to quantify the possible effects of multiple failures and CMFs on systems employing well-known CED schemes. The main conclusions of this paper are: (1) Our simulation results on

benchmark circuits reveal that we obtain marginal reduction in logic area by using CED schemes based on parity prediction instead of duplication; (2) CED schemes based on Berger codes and Bose-Lin codes incur very high logic area overhead; (3) For the simulated designs, diverse duplex systems with different implementations of the same logic function have a significant advantage over other CED schemes in providing protection against multiple failures and CMFs. This advantage makes diverse duplex systems a prominent candidate for implementing concurrent error detection in dependable systems. This result supports many of the observations in [Sedmak 78]. Looking at the future, research efforts must focus on cost-effective ways of designing CED techniques based on diverse duplication to reduce their area overhead while obtaining significant protection against multiple failures and CMFs.

9. Acknowledgments

This work was supported by Defense Advanced Research Projects Agency (DARPA) under Contract No. DABT63-97-C-0024. The authors wish to thank Prof. Nur Touba of Univ. of Texas at Austin and Nirmal Saxena, Philip Shirvani and Robert Huang of Stanford CRC.

10. References

[Aksenova 75] Aksenova, G. P. and E. S. Sogomonyan, "Design of Self-Checking Built-In Check Circuits for Automata with Memory," *Automation and Remote Control*, Vol. 36, pp. 1169-1177, July 1975.

[Avizienis 71] Avizienis, A., "Arithmetic Error Codes: Cost and Effectiveness Studies for Application in Digital System Design," *IEEE Trans. Computers*, Vol. C-20, No. 11, pp. 1322-1331, Nov. 1971.

[Avizienis 84] Avizienis, A. and J. P. J. Kelly, "Fault Tolerance by Design Diversity: Concepts and Experiments," *IEEE Computer*, pp. 67-80, Aug. 1984.

[Berger 61] Berger, J. M., "A Note on Error Detection Codes for Asymmetric Channels," *Information and Control*, Vol. 4, pp. 68-73, 1961.

[Bose 85] Bose, B. and D. J. Lin, "Systematic Unidirectional Error-Detecting Codes," *IEEE Trans. Comp.*, pp. 1026-1032, Nov. 1985.

[Chen 92] Chen, C. L., *et al.*, "Fault-tolerance Design of the IBM Enterprise System/9000 Type 9021 Processors," *IBM Journal Res. and Dev.*, Vol. 36, No. 4, pp. 765-779, July 1992.

[Das 98] Das, D. and N. A. Touba, "Synthesis of Circuits with Low-Cost Concurrent Error Detection based on Bose-Lin codes," *VLSI Test Symp.*, pp. 309-315, 1998.

[De 94] De, K., C. Natarajan, D. Nair and P. Banerjee, "RSYN: A System for Automated Synthesis of Reliable Multilevel Circuits," *IEEE Trans. VLSI*, Vol. 2, pp. 186-195, June 1994.

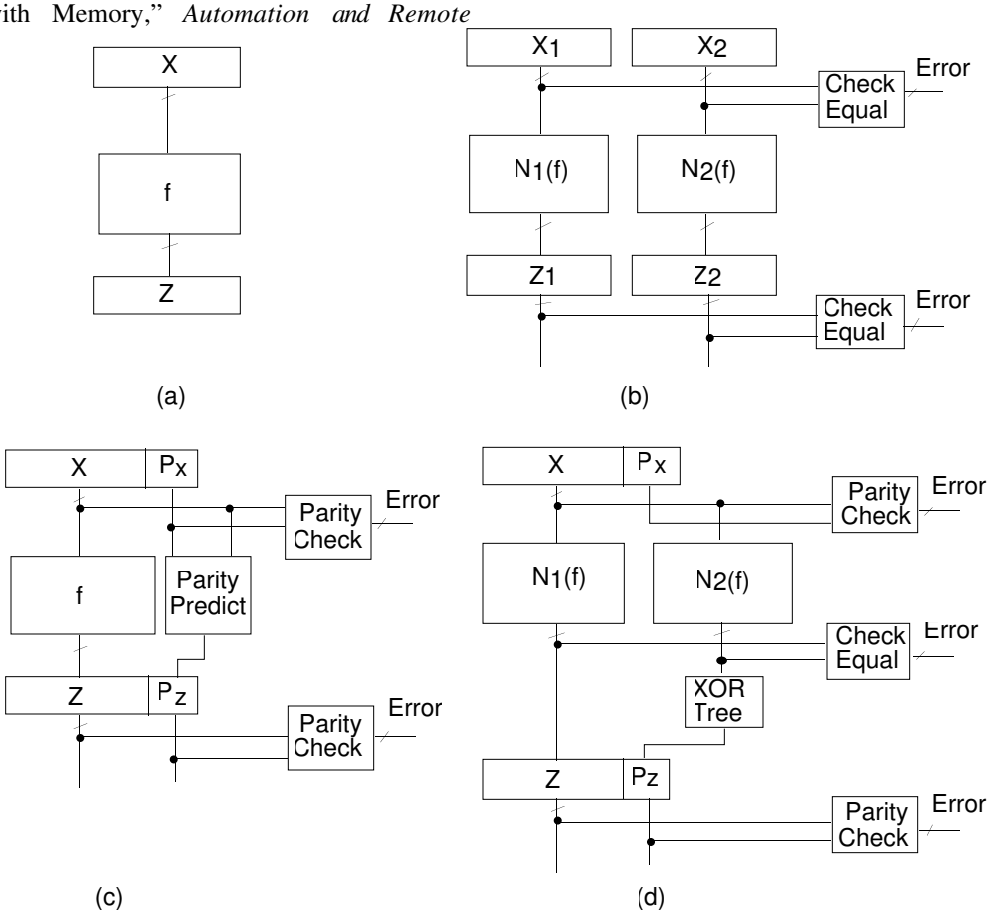

Figure 7.1. Systems with CED: (a) Example (b) Identical or Diverse Duplication (c) Parity prediction (d) Diverse duplication for combinational logic; parity prediction for registers and bus

[Hayes 76] Hayes, J. P., "Transition Count Testing of Combinational Logic Circuits," *IEEE Trans Computers,* Vol. C-25, No. 6, pp. 613-620, June 1976.

[Huang 00] Huang, W., N. R. Saxena and E. J. McCluskey, "A Reliable LZ Data Compressor on Reconfigurable Coprocessors," *Proc. IEEE Symp. Field Programmable Custom Computing Machines*, 2000.

[Hsiao 81] Hsiao, M-Y, W. C. Carter, J. W. Thomas and W. R. Stringfellow, "Reliability, Availability and Serviceability of IBM Computer Systems: A Quarter Century of Progress," *IBM Journal of Research and Development*, Vol. 25, No. 5, pp. 453-469, Sept. 1981.

[Jha 91] Jha, N. K., "Totally Self-Checking Checker Designs for Bose-Lin, Bose, and Blaum Codes," *IEEE Trans. CAD,* Vol. 10, No. 1, pp. 136-143, Jan. 1991.

[Jha 93] Jha, N. K. and S. J. Wang, "Design and Synthesis of Self-Checking VLSI Circuits," *IEEE Trans. CAD*, Vol. 12, pp. 878-887, June 1993.

[Jou 88] Jou, J-Y, and J. A. Abraham, "Fault-Tolerant FFT Networks," *IEEE Trans. Computers*, Vol. 37, No. 5, pp. 548-561, May 1988.

[Kraft 81] Kraft, G. D. and W. N. Toy, *Microprogrammed Control and Reliable Design of Small Computers*, 1981.

[LSI 96] *G10-p Cell-Based ASIC Products Databook*, LSI Logic, May 1996.

[Lala 94] Lala, J. H. and R. E. Harper, "Architectural principles for safety-critical real-time applications," *Proc. of the IEEE*, vol. 82, no. 1, pp. 25-40, Jan. 1994.

[Langdon 70] Langdon, G. G. and C. K. Tang, "Concurrent Error Detection for Group Look-ahead Binary Adders," *IBM Journal Res. and Dev.*, pp. 563-573, Sept. 1970.

[Mahmood 84] Mahmood, A., D. M. Andrews and E. J. McCluskey, "Executable Assertions and Flight Software," *Proc. AIAA/IEEE Digital Avionics Systems, Conf.*, pp. 346-351, 1984.

[Marouf 78] Marouf, M. A. and A. D. Friedman, "Design of Self-checking Checkers for Berger Codes," *Proc. FTCS*, pp. 179-184, 1978.

[McCluskey 90] McCluskey, E. J., "Design techniques for Testable Embedded Error Checkers," *IEEE Computer*, Vol. 23, No. 7, pp. 84-88, July 1990.

[Mitra 99a] Mitra, S., N. R. Saxena and E. J. McCluskey, "A Design Diversity Metric and Reliability Analysis for Redundant Systems," *Intl. Test Conf.*, pp. 662-671, 1999.

[Mitra 99b] Mitra, S., N. R. Saxena and E. J. McCluskey, "A Design Diversity Metric and Analysis of Redundant Systems," *Technical Report, Center for Reliable Computing, CRC-TR 99-4*, Stanford University, 1999.

[Mitra 00a] Mitra, S., N. R. Saxena and E. J. McCluskey, "Common-Mode Failures in Redundant VLSI Systems: A Survey," *IEEE Trans. Reliability*, 2000, To appear.

[Mitra 00b] Mitra, S. and E. J. McCluskey, "Combinational Logic Synthesis for Diversity in Duplex Systems," *Proc. Intl. Test Conf.*, 2000.

[Nicolaidis 93] Nicolaidis, M., "Efficient Implementations of Self-Checking Adders and ALUs," *Proc. Intl. Symp. Fault-Tolerant Computing*, pp. 586-595, 1993.

[Nicolaidis 97] Nicolaidis, M., R. O. Duarte, S. Manich and J. Figueras, "Fault-secure Parity Prediction Arithmetic Operators," *IEEE Design and Test of Computers*, Vol. 14, No. 2, pp. 60-71, 1997.

[Patel 82] Patel, J. H. and L. Y. Fung, "Concurrent Error Detection in ALUs by Recomputing with Shifted Operands," *IEEE Trans. Computers*, Vol. C-31, No. 7, pp. 589-595, July 1982.

[Pradhan 91] Pradhan, D. K., and S. K. Gupta, "A New Framework for Designing and Analyzing BIST Techniques and Zero Aliasing Compression," *IEEE Trans. Computers*, Vol. 40, No. 6, pp. 743-763, 1991.

[Pradhan 96] Pradhan, D. K., *Fault-Tolerant Computer System Design*, Prentice Hall, 1996.

[Ramamoorthy 75] Ramamoorthy, C. V. and Y-W Han, "Reliability Analysis of Systems with Concurrent Error Detection," *IEEE Trans. Computers*, Vol. C-24, No. 9, pp. 868-878, Sept. 1975.

[Rao 89] Rao, T. R. N. and E. Fujiwara, *Error-Control Coding for Computer Systems*, Prentice-Hall, 1989.

[Saxena 97] Saxena, N. R., and E. J. McCluskey, "Parallel Signature Analysis Design with Bounds on Aliasing," *IEEE Trans. Computers*, Vol. 46, No. 4, pp. 425-438, April 1997.

[Sedmak 78] Sedmak, R. M. and H. L. Liebergot, "Fault-Tolerance of a General-Purpose Computer Implemented by Very Large Scale Integration," *Proc. FTCS*, pp. 137-143, 1978.

[Sellers 68] Sellers, F., M-Y Hsiao and L. W. Bearnson, *Error Detection Logic for Digital Computers*, McGraw-Hill Book Company, 1968.

[Sentovich 92] Sentovich, E. M., *et al.*, "SIS: A System for Sequential Circuit Synthesis," *ERL Memo. No. UCB/ERL M92/41*, EECS, UC Berkeley, CA 94720.

[Shedletsky 78] Shedletsky, J.J., "Error Correction by Alternate-Data Retry," *IEEE Trans. Computers*, pp. 106-112, Feb. 1978.

[Siewiorek 92] Siewiorek, D. P. and R. S. Swarz, *Reliable Computer Systems: Design and Evaluation*, Digital Press, 1992.

[Spainhower 99] Spainhower, L. and T. A. Gregg, "S/390 Parallel Enterprise Server G5 fault tolerance," *IBM Journal of Research Development*, Vol. 43, pp. 863-873, Sept./Nov. 1999.

[Touba 97] Touba, N. A. and E. J. McCluskey, "Logic Synthesis of Multilevel Circuits with Concurrent Error Detection," *IEEE Trans. CAD*, Vol. 16, pp. 783-789, July 1997.

[Wakerly 78] Wakerly, J., *Error Detecting Codes, Self-checking Circuits and Applications*, 1978.

[Webb 97] Webb, C. F., and J. S. Liptay, "A High Frequency Custom S/390 Microprocessor," *IBM Journal Res. and Dev.*, Vol. 41, No. 4/5, pp. 463-474, 1997.

[Zeng 99] Zeng, C., N. R. Saxena and E. J. McCluskey, "Finite State Machine Synthesis with Concurrent Error Detection," *Proc. Intl. Test Conf.*, pp. 672-680, 1999.