

Center for Reliable Computing

TECHNICAL REPORT

A Design Diversity Metric and Analysis of Redundant Systems

Subhasish Mitra, Nirmal R. Saxena and Edward J. McCluskey

<p>99-4 Preliminary Version (CSL TR # 799) September, 1999</p>	<p>Center for Reliable Computing Gates Building 2A, Room 236 Computer Systems Laboratory Dept. of Electrical Engineering and Computer Science Stanford University Stanford, California 94305-9020</p>
<p>Abstract:</p> <p>Design diversity has long been used to protect redundant systems against common-mode failures. The conventional notion of diversity relies on “independent” generation of “different” implementations. This concept is qualitative and does not provide a basis to compare the reliabilities of two diverse systems. In this paper, for the first time, we present a metric to quantify diversity among several designs. Based on this metric, we derive analytical reliability models that show a simple relationship among design diversity, system failure rate, and mission time. We also perform availability analysis of redundant systems using our metric. In addition, we present simulation results to demonstrate the effectiveness of design diversity in duplex systems. For common-mode failures and design faults, there is a significant gain in using different implementations — however, as our analysis shows, the gain diminishes as the mission time increases. For independent multiple-module failures, we show that, mere use of different implementations does not always guarantee higher reliability compared to redundant systems with identical implementations — it is important to analyze the reliability of redundant systems using our metric. Our simulation results also demonstrate the usefulness of diversity in enhancing the self-testing properties of redundant systems.</p>	
<p>Funding:</p> <p>This work was supported by the Advanced Research Projects Agency under prime contract No. DABT63-97-C-0024.</p>	

Imprimatur: Philip Shirvani and Santiago Fernandez-Gomez

PRELIMINARY VERSION

A Design Diversity Metric and Analysis of Redundant Systems

Subhasish Mitra, Nirmal R. Saxena and Edward J. McCluskey

CRC Technical Report No. 99-4
(CSL TR No. ??)
May 1999

Center for Reliable Computing

Computer Systems Laboratory

Departments of Electrical Engineering and Computer Science

Stanford University, Stanford, California 94305

Abstract

Design diversity has long been used to protect redundant systems against common-mode failures. The conventional notion of diversity relies on “independent” generation of “different” implementations. This concept is qualitative and does not provide a basis to compare the reliabilities of two diverse systems. In this paper, for the first time, we present a metric to quantify diversity among several designs. Based on this metric, we derive analytical reliability models that show a simple relationship among design diversity, system failure rate, and mission time. We also perform availability analysis of redundant systems using our metric. In addition, we present simulation results to demonstrate the effectiveness of design diversity in duplex systems. For common-mode failures and design faults, there is a significant gain in using different implementations — however, as our analysis shows, the gain diminishes as the mission time increases. For independent multiple-module failures, we show that, mere use of different implementations does not always guarantee higher reliability compared to redundant systems with identical implementations — it is important to analyze the reliability of redundant systems using our metric. Our simulation results also demonstrate the usefulness of diversity in enhancing the self-testing properties of redundant systems

TABLE OF CONTENTS

1. Introduction	1
2. Design Diversity Metric and Reliability Analysis	3
3. Example	12
4. A Simulation-Based Approach	13
5. Self-Testing Property	17
6. Diversity Advantages in Configurable Systems	18
7. Conclusions	20
8. Acknowledgments	21
9. References	22
10. Appendix 1: Reliability Analysis	24

LIST OF FIGURES

Figure 1.1. A Duplex redundant system	1
Figure 2.1 A discrete time model of the system	5
Figure 2.2. Faults affecting modules simultaneously	5
Figure 2.3. Fault-Secure probability of duplex systems	7
Figure 2.4. Fault-secure probability of duplex systems (common-mode failures)	8
Figure 2.5. Effect of diversity with mission time	8
Figure 2.6. Markov chain for availability analysis	11
Figure 2.7. Comparison of availability of duplex systems	12
Figure 3.1. An example logic circuit	13
Figure 6.1. Reconfigurable computing test-bed	18
Figure 6.2. Results from experiments on a configurable computing test-bed	19
Figure 6.3. Results from experiments on a configurable computing test-bed	20

LIST OF TABLES

Table 2.1.	Behavior of faulty multiple-output circuits	4
Table 4.1.	Characteristics of simulated designs	14
Table 4.2.	Simulation 1 results	15
Table 4.3.	Simulation 2 results	16
Table 5.1.	Self-testing properties of diverse and non-diverse duplex systems	18

PRELIMINARY VERSION

1. INTRODUCTION

The use of redundancy techniques for designing systems with high data-integrity and availability has been studied extensively [Siewiorek 92][Pradhan 96]. A duplex system in the form of a self-checking pair is an example of a classical redundancy scheme (Fig. 1.1). As long as only one module fails, the system either produces correct results or indicates error situation.

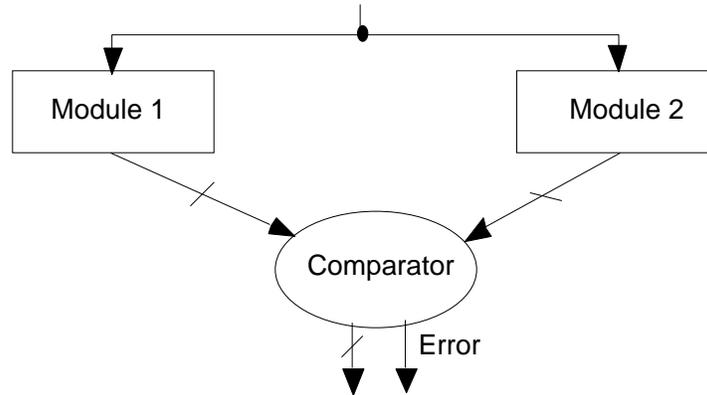


Figure 1.1. A Duplex Redundant System

In a redundant system, *common-mode failures* (CMFs) result from failures that affect more than one module at the same time, generally due to a common cause. These include operational failures that may be due to external (such as EMI, power-supply disturbances and radiation) or internal causes. In addition to these common-mode failures, with the increasing complexity of the various designs, design mistakes are becoming very significant. It has been pointed out in [Avizienis 84], although the use of redundant copies of hardware has proven to be quite effective in the detection of physical faults and subsequent system recovery, design faults are reproduced when redundant copies are made. Simple replication fails to enhance the system reliability against design faults.

Design diversity has been proposed in the past to protect redundant systems against common-mode failures. In [Avizienis 84], *design diversity* was defined as the independent generation of two or more software or hardware elements (e.g., program modules, VLSI circuit masks, etc.) to satisfy a given requirement. Design diversity was also proposed in [Lala 94] as an avoidance technique against common-mode failures. Design diversity has been applied to both software and hardware systems. *N*-version programming [Avizienis 77, Lyu 91] is used to achieve diversity in software systems. Hardware design diversity has been used in the Primary Flight Computer (PFC) system of Boeing 777 [Riter 95] and many other commercial systems [Briere 93]. For the Boeing 777, three different processors (from AMD, Intel and Motorola) are used in the

PRELIMINARY VERSION

PFC. Tohma proposed to use the implementations of logic functions in true and complemented forms during duplication [Tohma 71]. The use of a particular circuit and its dual was proposed in [Tamir 84] to achieve diversity in order to handle common-mode failures. The basic idea is that, with different implementations, common failure modes will probably cause different error effects.

Design diversity can prove to be useful in the context of dependable *Adaptive Computing Systems* (ACS). The field programmability of Field Programmable Gate Arrays (FPGAs) can be utilized to achieve diversity among the different modules. In an ACS environment, we can create diversity by synthesizing and downloading different implementations into FPGAs at any time. Thus, there is no need to manufacture multiple diverse ASICs. In order to quantify the effect of diversity on the reliability of a redundant system, a metric is needed to *quantify diversity* among designs with the same specification [Tamir 84].

In addition to common-mode failures, with the high density of logic gates in a VLSI chip, multiple failures may become more frequent. For example, current research shows that multiple-event upsets (possibly due to a single radiation source) are common in VLSI chips [Liu 97][Reed 97]. The classical reliability models of redundant systems are pessimistic because, in the presence of multiple module failures, they do not consider compensating effects of different faults [Siewiorek 75]. It is interesting to find out whether design diversity also helps in achieving better compensating effects of different faults, compared to simple replication.

In this paper, we address problems related to design diversity and examine their effects on the reliability of a redundant system. Some preliminary ideas related to this work were reported in [Saxena 98] and [Mitra 99]. Our main contributions are: (1) developing a metric to quantify diversity among several designs; and (2) using this metric to perform reliability and availability analysis of redundant systems. In Sec. 2, we introduce a design diversity metric and perform reliability and availability analysis of redundant systems using this metric. Section 3 presents some preliminaries related to the stuck-at fault model and illustrates our analysis with the help of an example. We present simulation results in Sec. 4. Section 5 examines the effect of design diversity on the self-testing properties of a duplex system. We present experimental results demonstrating the advantages of using design diversity in configurable systems in Sec. 6. Finally, we conclude in Sec. 7.

2. Design Diversity Metric And Reliability Analysis

2.1. *D*: A Design Diversity Metric

In this section, we introduce a metric to quantify diversity among several designs. We define the metric for a system with two designs implementing the same function. The metric has application in estimating the reliability of NMR systems with masking redundancy. Before defining the diversity metric, we first define the notion of diversity between two implementations with respect to a fault pair.

For two designs implementing the same function, the *diversity with respect to a fault pair* (f_i, f_j) , $d_{i,j}$, is the probability that the designs do not produce identical error patterns, in response to a given input sequence, when f_i and f_j affect the first and the second implementations, respectively.

For a given fault model, the *design diversity metric*, D , between two designs is the expected value of the diversity with respect to different fault pairs. Mathematically, we have $D = \sum_{(f_i, f_j)} P(f_i, f_j) d_{i,j}$, where $P(f_i, f_j)$ is the probability of the fault pair (f_i, f_j) .

D is the probability that, in response to a given input sequence, the two implementations either produce error-free outputs or produce *different* error patterns on their outputs.

Example: Consider any combinational logic function with n inputs and a single output. The fault model considered is such, that a combinational circuit remains combinational in the presence of the fault. Let us consider two implementations (N_1 and N_2) of the given combinational logic function.

The *joint detectability*, $k_{i,j}$, of a fault pair (f_i, f_j) is the number of input patterns that detect both f_i and f_j . This definition follows from the idea of detectability developed in [McCluskey 88].

If we assume that all the input patterns are equally likely, then we can write $d_{i,j} = 1 - \frac{k_{i,j}}{2^n}$.

The $d_{i,j}$'s generate a diversity profile for the two implementations with respect to a fault model. Consider a duplex system consisting of the two implementations under consideration. In response to any input combination, the implementations can produce one of the following cases at their outputs. (1) Both of them produce correct outputs. (2) One of them produces correct output and the other produces incorrect output. (3) Both of them produce the same incorrect value.

For the first case, the duplex system will produce correct outputs. For the second case, the system will report a mismatch so that appropriate recovery actions can be taken. However, for the third case, the system will produce an incorrect output without reporting

PRELIMINARY VERSION

a mismatch — thus, for the third case, the *integrity* of the system is lost due to the presence of faults in the two implementations. In the literature on fault-tolerance [Siewiorek 92][Pradhan 96], this system integrity has been referred to as the *fault-secure* property.

The quantity $d_{i,j}$ is the probability that a duplex system, having two implementations of the logic function under consideration, is fault-secure when faults f_i and f_j affect the first (N_1) and the second (N_2) implementations, respectively.

If we assume that all fault pairs are equally probable and there are m fault pairs (f_i, f_j), then the D metric for the two implementations is: $D = \frac{1}{m} \sum_{i,j} d_{i,j}$.

We extend the above example to consider multiple-output combinational logic circuits. For a fault pair (f_i, f_j) affecting the two implementations, we define k_{ij} as the number of input patterns, in response to each of which, both the implementations produce the same erroneous output pattern. We can use the same formulas as the single output case.

Table 2.1. Behavior of faulty multiple output circuits

Inputs	Fault-free outputs	Faulty outputs (Implementation 1)	Faulty outputs (Implementation 2)
00	0 1	0 0	1 0
01	1 0	1 0	1 0
10	0 0	1 0	1 0
11	1 1	1 0	1 0

For example, consider a combinational logic function with two inputs and two outputs (Table 2.1). Suppose that, faults f_i and f_j affect the first and the second implementations, respectively. The responses of the two implementations in the presence of the faults are shown in Table 2.1. The faulty output bits are highlighted in the third and fourth columns of Table 2.1. It is clear that for the calculation of k_{ij} , we have to consider only the input patterns 10 and 11.

The above illustration of the design diversity metric can also be extended to sequential circuits and software programs. For small or medium-sized systems, the exact value of the diversity metric can be calculated manually or using computer programs. For large systems, the value can be estimated by using simulation techniques.

For two identical implementations of the same function, a common-mode failure (e.g., a design mistake) can be modeled as the same fault f_i affecting the two implementations. Let m be the number of input sequences for which these two implementations produce identical error patterns at the outputs. If the second implementation is different from the first, for any fault f_j affecting the second

implementation (and f_i affecting the first implementation), we cannot have more than m input sequences that produce identical error patterns at the outputs of the two implementations. Hence, $d_{i,i} \neq d_{i,j}$. This property is useful for enhancing the reliability of a redundant system against common-mode failures by using diversity.

2.2. Reliability Analysis

In this section, we calculate the reliability of duplex systems using the diversity metric described in Sec. 2.1. We define the *reliability of a duplex system* as the probability that the system is fault-secure. The reliability calculation is independent of whether the redundant components are exact replicas or different implementations. We assume a discrete time model for the system. In such a model, the time axis is broken up into discrete time cycles and we apply inputs and observe outputs only at cycle boundaries.

As shown in Fig. 2.1, input combination (vector) v_i is applied at the beginning of the i^{th} cycle. Also, in Fig. 2.1, the first system becomes faulty (f_1) during cycle i and the second system becomes faulty (f_2) during cycle j . Let p be the probability that a particular module is affected by a fault at any cycle. For simplicity, we assume that this probability p is the same for all the modules in the system at all times. The probability p can be looked upon as the failure rate per cycle.

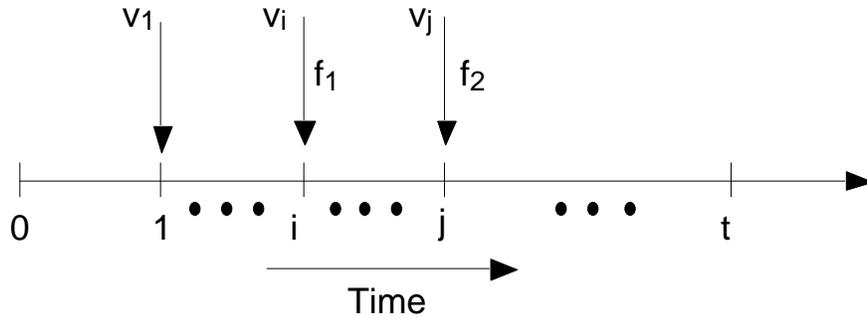


Figure 2.1. A discrete time model of the system

For a given fault pair (f_1, f_2) , there are two possible cases. In the first case, both the faults appear in the same cycle. The situation is shown in Fig. 2.2.

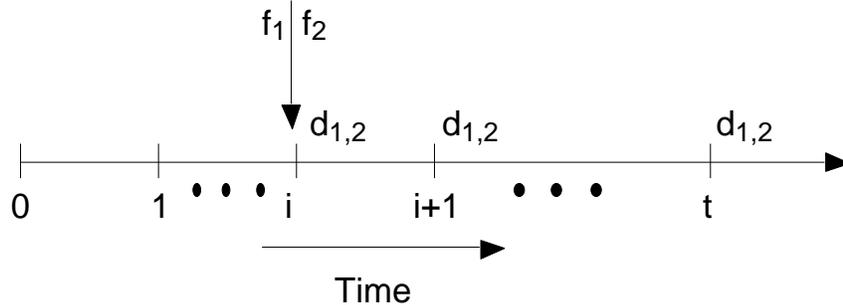


Figure 2.2. Faults affect the modules simultaneously

PRELIMINARY VERSION

In Fig. 2.2, faults f_1 and f_2 affect modules 1 and 2, simultaneously at cycle i . It may be argued that if a random fault appears in a particular module, then chances are high that the second fault will also appear in that same module. However, we do not assume any such correlation in this paper. At time 0, everything is fault-free. So, before cycle i the system will produce correct results. However, starting from time i , in each cycle, the system will produce correct results with the probability equal to $d_{1,2}$. The probability $s_1(f_1, f_2, t)$ that the system is fault-secure up to time t , even in the presence of the two faults f_1 and f_2 , is given by:

$$s_1(f_1, f_2, t) = p^2 d_{1,2} \frac{[d_{1,2}^t - (1-p)^{2t}]}{[d_{1,2} - (1-p)^2]}$$

The derivation of the above expression is shown in the appendix. Next, we consider the case where f_1 and f_2 appear at different cycles.

As discussed earlier, in Fig. 2.1, Module 1 becomes faulty during cycle i and Module 2 becomes faulty during cycle j . It is clear that up to time j , a duplex system will be fault-secure. Hence, starting from time j , the system will be fault-secure with probability $d_{1,2}$. Thus, the probability $s_2(f_1, f_2, t)$ that the system is fault-secure up to time t , in the presence of the two faults f_1 and f_2 is given by the following equation.

$$s_2(f_1, f_2, t) = \frac{2}{(d_{1,2} - 1 + p)} (1-p)p^2 d_{1,2}^2 \frac{[d_{1,2}^{t-1} - (1-p)^{2t-2}]}{[d_{1,2} - (1-p)^2]} - \frac{2}{(d_{1,2} - 1 + p)} (1-p)^t p d_{1,2} [1 - (1-p)^{t-1}]$$

The derivation of the second case is also shown in the appendix. This case is more complicated than the first case and is useful when we consider random independent faults in multiple modules. We have:

$$s(f_1, f_2, t) = s_1(f_1, f_2, t) + s_2(f_1, f_2, t)$$

Here $s(f_1, f_2, t)$ is the probability that a duplex system is fault-secure up to time t , when Module 1 is affected by fault f_1 and Module 2 by fault f_2 .

We can characterize a duplex system using our diversity metric. In the following calculations, we assume that once a module becomes faulty, no other fault appears in that module. This assumption is simplistic and allows us to obtain closed-form reliability expressions. We calculate the probability that, up to time t , a duplex system is fault-secure. It is given by the following expression:

$$(1-p)^{2t} + 2(1-p)^t [1 - (1-p)^t] + \underset{f_1, f_2}{P(f_1, f_2)} s(f_1, f_2, t)$$

The above expression follows from the fact that, in a duplex system, when none of the modules fails the system produces correct outputs. When only one of the modules

fails (due to single or multiple faults), the system is fault-secure. When both modules are faulty, then we have to consider the $d_{1,2}$ value for the fault pair (f_1, f_2) in the two modules. $P(f_1, f_2)$ is the probability that faults f_1 and f_2 appear in modules 1 and 2, respectively.

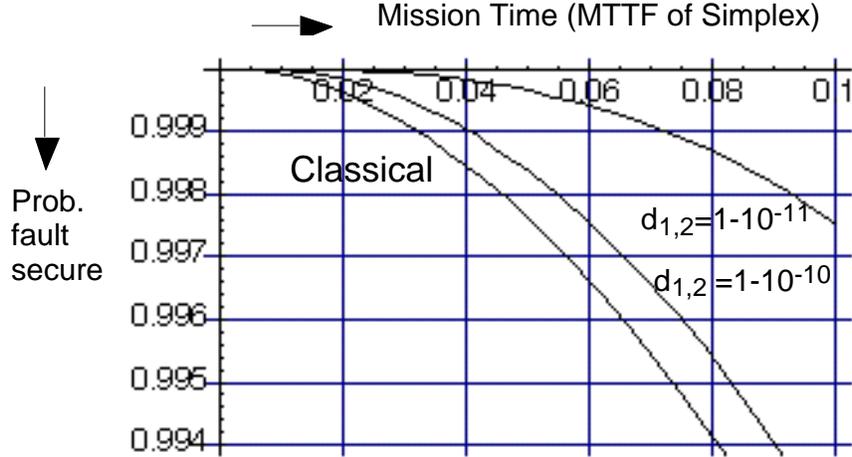


Figure 2.3. Fault-secure probability of a duplex system with multiple independent failures

In Fig. 2.3, for a given pair of faults (f_1, f_2) , we show the plots of the above expression for different values of $d_{1,2}$. The mission time is shown along the X-axis — the MTTF (Mean Time To Failure in cycles) of a simplex system corresponds to 1 time unit. The probability that a fault appears in one cycle is 10^{-12} . Along the Y-axis, we show the probability that the duplex system is fault-secure. The classical analysis of duplex systems is pessimistic since it assumes that the system ceases to be fault-secure when two modules are faulty.

The above expressions can be modified for common-mode failures (CMF). The probability that a duplex system is fault-secure against common-mode failures up to time t , is given by the following expression:

$$(1-p)^t + \sum_{f_1, f_2} P(f_1, f_2) z(f_1, f_2, t)$$

Here, p is the probability that a CMF affects the two modules. In the above expression, $z(f_1, f_2, t)$ is given by the following formula:

$$z(f_1, f_2, t) = pd_{1,2} \frac{[d_{1,2}^t - (1-p)^t]}{[d_{1,2} - (1-p)]}$$

The above expression is maximized when $d_{1,2}$ is of the order of $(1-p)$. This suggests that, for a common-mode failure that can be modeled as fault pair (f_1, f_2) , we can obtain appreciable reliability improvement over classical systems when the value of $d_{1,2}$ is of the order of $(1-p)$. The following observations can be derived from this

relationship.

9. When the failure rate is high, even a small diversity can help enhance the system reliability over traditional replication.

10. If the failure rate is low, then $d_{1,2}$ must be extremely high for appreciable reliability improvement over classical systems. As a limiting case, consider the situation when the CMF failure rate is 0. In that case, diversity will not buy us any extra reliability against CMFs.

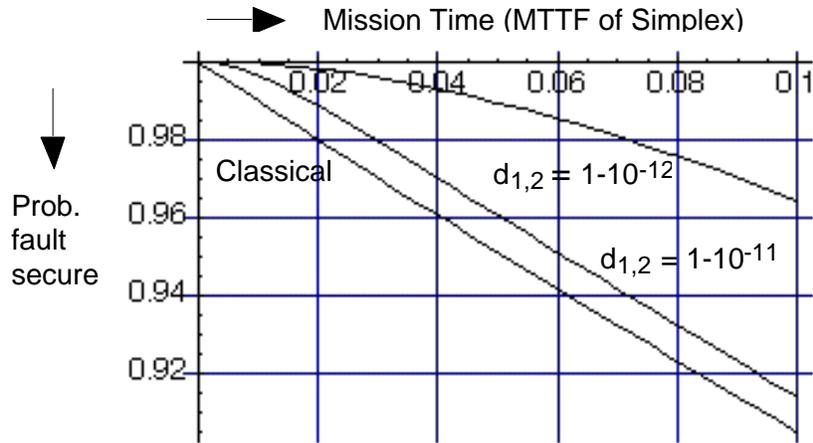


Figure 2.4. Fault-secure probability of a duplex system against common-mode failures

In Fig. 2.4, for a given pair of faults (f_1, f_2), we show the plots of the above fault-secure probability expression for the different values of $d_{1,2}$. The failure rate per cycle is 10^{-13} . It is clear that we get appreciable improvement in reliability (over classical systems) when the value of $d_{1,2}$ is very high ($1-10^{-12}$ or more). When the value of $d_{1,2}$ is less than $1-10^{-12}$, we do not see high reliability improvement.

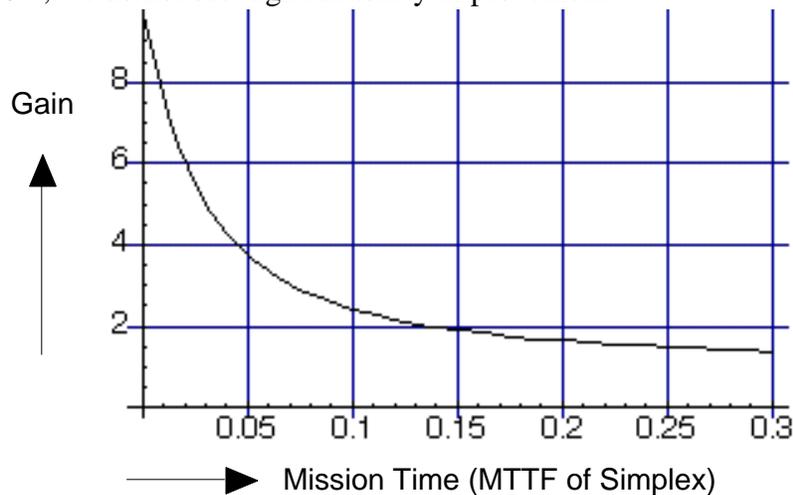


Figure 2.5. Effect of diversity with mission time (for common-mode failures)

PRELIMINARY VERSION

In Fig. 2.5, we show how the reliability improvement obtained from diversity depends on mission time. On the Y-axis of the graph in Fig. 2.5, we plot the ratio of the following two quantities.

9. The probability that a duplex system is not fault-secure at time i , for a fault pair (f_1, f_2) with $d_{1,2} = 1-10^{-11}$.
10. The probability that a duplex system is not fault-secure at time i , for fault pair (f_1, f_2) with $d_{1,2} = 1-10^{-12}$. The failure rate per cycle is 10^{-13} .

We call this ratio the *gain*. On the X-axis, we plot the mission time. As Fig. 2.5 shows, the gain diminishes with longer mission times. This analysis allows us to derive relationships between the reliability of a redundant system, the diversity incorporated to protect the system against common-mode failures and the mission time. The relationship between diversity and mission time can also be used to determine checkpoint intervals in a redundant system. For example, referring to Fig. 2.5, we can checkpoint the state of the system when the gain is close to 1. Thus, our design diversity metric is a very fundamental property and can be used to understand different trade-offs associated with the design of dependable systems using redundancy.

Next, we estimate the error latency using our design diversity metric. Consider a duplex system with two implementations N_1 and N_2 of the same logic function. Let us suppose that the faults f_1 and f_2 affect the two implementations at cycle c . The *error latency* is defined to be the number of cycles from c after which both the implementations produce the same error pattern at the output. For more discussions on error latency, the reader is referred to [Shedletsky 76]. The probability that the error latency is t ($t > 0$) is given by: $d_{1,2}^{t-1}(1-d_{1,2})$. Here, the assumption is that $d_{1,2}$ value is strictly less than 1. If the $d_{1,2}$ value is equal to 1, then the error latency is always equal to T , the mission time. The expected error latency is given by the following formula:

$$E_{t,(f_1,f_2),d_{1,2}} = \frac{P(f_1, f_2) t d_{1,2}^{t-1} (1 - d_{1,2})}{1} + \frac{P(f_1, f_2) T}{(f_1, f_2), d_{1,2} = 1}$$

From this expression, it is clear that for long mission times (i.e., large values of t), the probability value approaches to 0 when the $d_{1,2}$ value for the fault pair is less than 1. Thus, the fault pairs which have their $d_{i,j}$ values equal to 1 (i.e., the compensating fault pairs) play a dominant role in determining the error latency for long mission times. Hence, the value of the expected error latency is determined by the percentage of compensating fault pairs. Simplification of the above expression produces the following expression for the expected latency of a duplex system in terms of the diversity metrics with respect to the different fault pairs.

PRELIMINARY VERSION

$$\text{Expected error latency} = \sum_{(f_1, f_2), d_{1,2}} \frac{P(f_1, f_2)}{(1 - d_{1,2})} + \sum_{(f_1, f_2), d_{1,2}=1} P(f_1, f_2)T$$

Consider the case of design mistakes that are special cases of common-mode failures. For these cases, the fault is *always* present. Simple analysis reveals that the probability that a duplex system is fault-secure up to time t , in the presence of design mistakes, is:

$$\sum_{f_1, f_2} P(f_1, f_2) d_{1,2}^t$$

Thus, for design mistakes, for a given fault pair (f_1, f_2) , the more the value of $d_{1,2}$, the more is the system reliability. This implies that, for design mistakes, diversity among the two implementations in a duplex system helps to increase the probability that the system is fault-secure.

While diversity in hardware designs is the main focus of this paper, the above ideas can be extended to analyze diversity in software modules. For estimating the diversity metric for software modules, we need to have a fault model for the software under consideration. Considering the range of values the input variables to the software module can possibly take, it may be difficult to compute the exact value of the metric. However, the value of the metric can be estimated using simulation techniques. Note that, our observations about the relationships between diversity, mission time and failure rate still hold for software systems. One key feature of our analysis technique is that, it is powerful, but at the same time, simpler than the models in [Eckhardt 85], [Tomek 93] and [Lyu 95].

2.3. Availability Analysis

In this section, we perform availability analysis of duplex systems with repair capabilities using our diversity metric. For the purpose of our analysis, we assume that p is the probability that a (common-mode) failure affects the system during a particular cycle. The failure can manifest as fault f_1 and f_2 affecting Module 1 and Module 2, respectively. In our analysis, we use the following quantities, as described below.

The metric $d_{1,2}$ is the probability that the two modules do not produce the same error pattern (at their outputs) in response to a given input sequence, when they are affected by the faults f_1 and f_2 .

We define another quantity, $t_{1,2}$, which is the probability that the two modules do not produce any error at their outputs in response to a given input sequence, when they are affected by the faults f_1 and f_2 .

The quantity $d_{1,2} - t_{1,2}$ is the probability that the two modules will produce non-

identical error patterns (at their outputs) in response to a given input sequence, when they are affected by the faults f_1 and f_2 .

The Markov chain used for our analysis is shown in Fig. 2.6. In the Markov chain, the system starts at the *Good* state. As long as a fault does not appear, the system remains in the *Good* state. However, as soon as a fault appears, the system goes to the *Faulty Correct* state. The probability that both the modules produce correct outputs, in spite of the presence of the fault, is $t_{1,2}$. The probability that the modules produce identical errors at their outputs is $1 - d_{1,2}$. Thus, with probability $d_{1,2} - t_{1,2}$, the modules produce non-identical erroneous — this means that the presence of the fault is detected. Once the fault is detected, the system enters the *Repair* state. We have assumed that the expected number of cycles required to repair the system is $\frac{1}{m}$. For modeling the repair

operation, we could as well use a repair rate. However, in the context of re-configurable systems, we can have bounds on repair time, which we can use during the above Markov analysis. The *availability* is given by the probability that the system is in the *Good* or the *Faulty Correct* state. In the following graph (Fig. 2.7), we show the dependence of availability on the values of $d_{1,2}$ and $t_{1,2}$. This analysis implications on the usefulness of diversity for enhancing the self-testing property and hence, the availability of duplex systems. The analysis can be extended for other redundant systems (e.g., NMR systems).

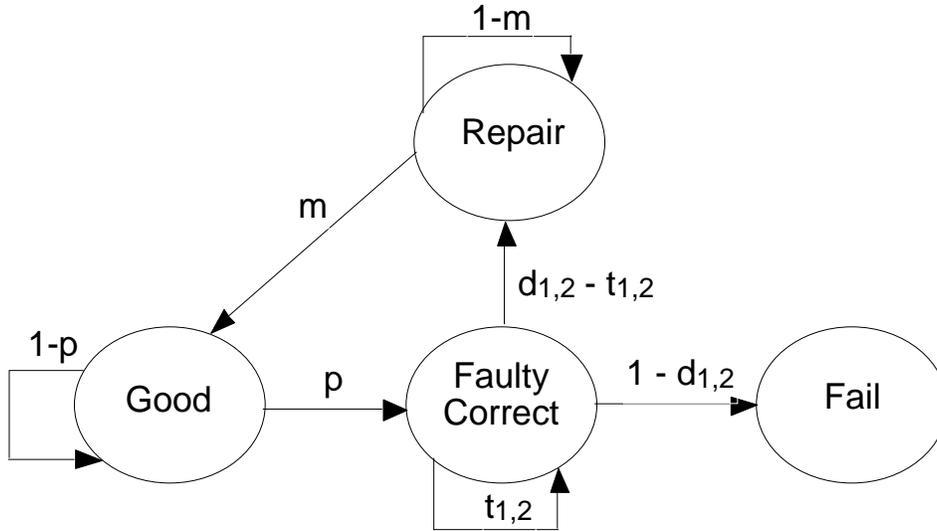


Figure 2.6. Markov Chain for availability analysis

In Fig. 2.7, we plot the availability values for two duplex systems. The probability (p) that a fault pair appears in a particular cycle is 10^{-8} . The number of repair cycles ($\frac{1}{m}$) is 100. Both the systems have the value of $d_{1,2}$ equal to $1 \cdot 10^{-5}$ for a fault pair. However, one of the systems (shown in Fig. 2.7) has the value of $t_{1,2}$ equal to $d_{1,2}$ and the

other one has the $t_{1,2}$ value equal to half of $d_{1,2}$. As can be seen in Fig. 2.7, initially the system having $t_{1,2} = d_{1,2}$ has a higher availability (since the probability that it stays in the *Faulty Correct* state is high). However, as time increases, the availability of the system with $t_{1,2} = 0.5*d_{1,2}$ decreases at a much smaller rate compared to the system with $t_{1,2} = d_{1,2}$. This is because, for the system with $d_{1,2}$ equal to $t_{1,2}$, there is no repair capability in contrast to the other system.

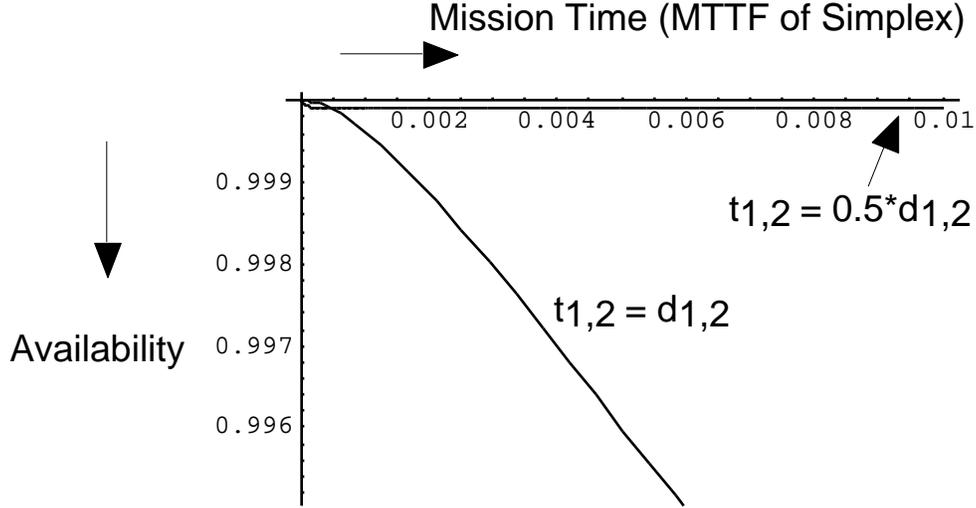


Figure 2.7. Comparison of the availability of duplex systems

We validate our observations using simulation data in Sec. 4. For simulation purposes, we used the stuck-at fault model. In the next section, we introduce the preliminaries related to the stuck-at fault model and illustrate the calculation of our diversity metric using an example.

3. Example

Research in the area of digital testing and diagnosis of combinational and sequential logic circuits has demonstrated the effectiveness of the logical *stuck-at* fault model. In this model, the failures in a logic circuit behave as if as some lines in the circuit assume *constant* logical values, either 1 or 0, independent of the logic values on other lines of the circuit.

For the rest of this paper, we assume that all failures manifest as stuck-at faults in the circuit. We also assume that the failures are permanent; i.e., if a stuck-at fault shows up at some time instant t , then the fault remains at all time instants greater than t . For circuits made from SRAM-based FPGAs, unless we re-initialize the SRAMs (reload a given configuration), a transient fault in the configuration SRAM persists. Thus, the assumption of the permanent fault behavior is reasonable.

PRELIMINARY VERSION

For example, consider the network shown in Fig. 3.1. The function implemented by the network is $wx + y$. Consider a stuck-at-0 (s-a-0) fault on the line y , denoted by $y/0$. The function implemented by the network, in the presence of the fault, is wx . Thus, $w = 1$, $x = 0$ and $y = 1$, when applied to the input of the logic circuit causes the faulty network to produce a 0 and the fault-free network to produce a 1. Therefore, the fault $y/0$ is detected by the pattern $w = 1, x = 0, y = 1$.

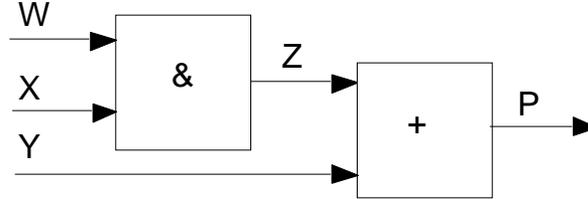


Figure 3.1. An example logic circuit

A fault is said to be functionally equivalent to another fault if and only if the output function realized by the network with only the first fault present is equal to the function realized when only the second fault is present. For example, in the network of Fig. 3.1, in the presence of the fault $x/0$, the function implemented is y . In the presence of the fault $z/0$, the function implemented by the network is also y . Hence, the faults $x/0$ and $z/0$ are functionally equivalent. The set of functionally equivalent faults forms an equivalence class. A fault f_1 *dominates* fault f_2 if and only if all input combinations that detect f_2 also detect f_1 . In our example, the fault $p/0$ dominates fault $z/0$. Techniques for obtaining equivalence and dominance relationships among different fault pairs have been described in [McCluskey 71] and [To 73].

We illustrate the calculation of our design diversity metric with respect to single stuck-at faults in the circuit of Fig. 3.1. There are 10 single-stuck faults associated with this network. The faults are: $w/0$, $w/1$, $x/0$, $x/1$, $y/0$, $y/1$, $z/0$, $z/1$, $p/0$ and $p/1$. The corresponding fault equivalence classes are: $F_1 = \{w/0, x/0, z/0\}$, $F_2 = \{y/1, z/1, p/1\}$, $F_3 = \{w/1\}$, $F_4 = \{x/1\}$, $F_5 = \{y/0\}$ and $F_6 = \{p/0\}$. The set of vectors that detect the faults in F_1 is $V_1 = \{w = 1, x = 1, y = 0\}$. We write $V_1 = \{110\}$. Similarly, $V_2 = \{000, 010, 100\}$, $V_3 = \{010\}$, $V_4 = \{100\}$, $V_5 = \{001, 011, 101\}$ and $V_6 = \{001, 011, 101, 111, 110\}$. Here, the number of inputs (n) is 3. Consider the fault pair $(f_1, f_2) = (w/0, p/0)$. The set of vectors that detect $w/0$ is V_1 . The set of vectors that detect $p/0$ is V_6 and $V_1 \cap V_6 = \{110\}$. Thus, the value of $d_{1,2}$ is $7/8$. In this way all the $d_{i,j}$'s and the D metric can be calculated.

4. A Simulation-Based Approach

As we noted earlier, it is difficult to model the entire complex system

PRELIMINARY VERSION

mathematically. Even with the stuck-at fault model, it is difficult to derive the exact reliability equation for the following reasons:

9. For a given pair of faults (f_1, f_2) , the calculation of $d_{1,2}$ is an NP-complete problem [Gary 79]. The problem is related to the NP-complete test generation problem.
10. If multiple stuck-at faults appear in the modules at different cycles, then the reliability expressions will become complicated. In fact, it may not be possible to obtain a closed form.

Hence, we developed a simulation environment to examine the reliability of a redundant system in the presence of multiple faulty modules.

Table 4.1. Characteristics of simulated designs

Circuit	# Inputs	# Outputs	# SSF (T)	# SSF (C)
Z5xp1	7	10	550	610
apex4	9	19	9636	8578
clip	9	5	698	664
inc	7	9	486	506
rd84	8	4	568	398

For generating *different* designs, we minimized the truth tables corresponding to some MCNC benchmark circuits (*clip*, *inc*, *Z5xp1*, *apex4* and *rd84*) using *espresso*. Then, we synthesized logic circuits after applying multi-level optimizations using the *rugged* script available in *sis* [Sentovich 92]. We subsequently mapped the multi-level logic circuits to the LSI Logic G-10p technology library [LSI 96]. Next, we complemented the outputs in the truth tables of the benchmark circuits to generate new truth tables. We used the same synthesis procedure for these new truth tables. Finally, we added inverters at the outputs of the new designs obtained. Table 4.1 summarizes the characteristics of the different simulated designs.

In the fourth column of Table 4.1, we report the number of candidate single stuck faults for the implementations of the circuits, obtained by synthesizing the given specification. The fifth column shows the number of candidate single stuck faults for the implementations of the circuits, obtained by synthesizing the given specifications with complemented outputs.

Simulation 1

For Simulation 1, for each benchmark circuit, we built duplex systems with identical and different implementations. For each of these systems, we performed 100,000 experiments. In each experiment, we randomly picked up a single stuck-at fault pair (f_1, f_2) such that the fault f_1 affects Module 1 and f_2 affects Module 2. We injected these faults into the modules, applied input patterns from a counter (with random seed)

and calculated the error latency (the number of cycles after which the system ceases to be fault-secure). The expected error latency for the injected fault pairs is shown in Table 4.2. We also calculated the percentage of fault pairs for which none of the two modules produced the same erroneous outputs at the same time (compensating fault pairs). These are the fault pairs (f_1, f_2) that have $d_{1,2}$ equal to 1.

Table 4.2. Simulation 1 results

Circuit Name	Copies	Error Latency (cycles)	% compensating fault pairs ($d_{i,j} = 1$)
Z5xp1	T, T	6733	66.96
	T, C	6869	68.76
apex4	T, T	8594	85.71
	T, C	8094	80.51
clip	T, T	7951	79.24
	T, C	7869	78.44
inc	T, T	7666	76.54
	T, C	7516	75.08
	C, C	7512	74.90
rd84	T, T	7638	76.23
	T, C	6797	67.73
	C, C	7051	70.40

As shown in Table 4.2, a duplex system consisting of different implementations of the *Z5xp1* circuit has a higher percentage of compensating fault pairs, compared to the non-diverse version — however, that is not generally true. For example, for the *clip* benchmark, the non-diverse duplex system has a higher percentage of compensating fault pairs. For compensating fault pairs, the error latency is strictly infinity — we assumed the value to be 10,000 cycles for our experiments. This is because, the number of inputs of the benchmark circuits under consideration lie between 7 and 9. Thus, the total number of input patterns is between 128 and 512. Note that, the expected error latency is dependent on the number of compensating fault pairs. This dependence of error latency on the number of compensating fault pairs has been explained earlier in Sec. 2.1.

In [Sakov 87], for a given combinational logic function, the fault detectability profiles for different implementations have been reported. Further studies are needed to synthesize circuit structures with high values of $d_{i,j}$ for different fault pairs. It has been proved in [To 73] that, for fanout-free combinational logic networks, all internal single stuck-at faults are either equivalent to or dominate single stuck-at faults on the primary inputs of the network. Thus, if we want to implement two diverse fanout-free networks implementing the same function, the $d_{i,j}$ values of the different fault pairs will be strongly dependent on the input combinations detecting the single stuck-at faults on network inputs and outputs. For both the networks, the set of patterns that detect the input or output stuck-at faults is independent of the network structure and is directly determined

PRELIMINARY VERSION

by the function the networks are implementing. Thus, chances are low that for fanout-free networks and stuck-at faults, the diversity metric is going to achieve appreciable high values for networks synthesized in different ways, compared to simple replication. Thus, it appears to be important to focus on achieving diverse fanout structures of different networks to obtain high values of the diversity metric for fault pairs.

Simulation 2

Our previous simulation results mainly focused on independent faults in multiple modules of a duplex system. However, it has been observed in the literature [Avizienis 84][Lala 94], that design diversity is useful for handling correlated failures and common-mode failures. Since we did not find any data on common-mode failure mechanisms, we performed the following sets of experiments to estimate the effect of diversity in the presence of common-mode failures.

Table 4.3. Simulation 2 results

Circuit Name	Duplex #	Copies	Worst Error Latency (cycles)
Z5xp1	1	T, T	10
	2	T, C	1711
	3	C, C	14
clip	5	T, T	35
	6	T, C	372
	7	C, C	48
inc	8	T, T	16
	9	T, C	1645
	10	C, C	17
rd84	11	T, T	35
	12	T, C	301
	13	C, C	21

In a duplicated system with identical implementations, we can find a one-to-one correspondence between the leads of the two copies. Hence, for these duplicated systems, we injected fault pairs (f_1, f_2) such that f_1 and f_2 affect lead i of Module 1 and Module 2, respectively. Note that, in the presence of f_1 and f_2 , the two modules behave exactly in the same way. Hence, they can be called common-mode faults. With this setup, we found the error latency for these common-mode faults. For duplex systems with different implementations, we cannot establish such a one-to-one correspondence between the leads of the two copies. Hence, for each fault f_1 in Module 1, we found the fault f_2 in Module 2 with the minimum value of $d_{1,2}$ using exhaustive simulation. Thus, for f_1 affecting Module 1, we have the least error latency when f_2 affects Module 2. Hence, the fault pair (f_1, f_2) is called the worst-case fault pair with the worst case latency. Then we averaged the worst-case latencies over all the worst-case fault pairs — this number is reported in the fourth column of Table 4.3.

The results in Table 4.3 show a distinct advantage of using different

implementations over non-diverse designs for common-mode faults. This is because, the worst case error latency of a common-mode fault in a duplex system with different implementations is at least an order of magnitude larger than the error latency of a common-mode fault in a duplex system with identical implementations.

In order to bring into perspective the significance of this increased error latency, we consider the execution of an application that uses the *Z5xp1* circuit of Table 4.3. If the mission time of the application is of the order of hundreds of cycles, then the system with two identical implementations will fail *in the presence of CMFs*. However, a system with two different implementations of *Z5xp1* will be able to finish the task, on an average, *in the presence of CMFs*. Finally, if the mission time is of the order of thousands of cycles, then in the presence of CMFs, none of these systems will be able to finish the task successfully. This result can also be explained from the properties of the diversity metric discussed in Sec. 2.1. The relationship of this result with the CMF rate is explained in Sec. 2.2.

Suppose that we have a system for which the common-mode failures affect only the inputs. In such a scenario, the systems with different implementations that we considered are *not diverse so far as the inputs are concerned*. Thus, such systems do not provide no extra protection against the common-mode failures of interest (affecting only the inputs) compared to systems with identical implementations. This argument motivates research in developing common-mode fault models and designing redundant systems with sufficient diversity against the modeled common-mode faults.

5. Self-testing Property

In this section, we discuss the possible effects of having design diversity on the self-testing property of a duplicated system. A duplicated system is called *self-testing* with respect to a fault pair (f_1, f_2) (f_1 affecting Module 1 and f_2 affecting Module 2) if and only if, there exists an input combination for which the two modules produce different outputs in the presence of the faults.

For the purpose of the experiment, we assume that the failures show up as single-stuck faults in each of the two modules under consideration. The self-testing property ensures that, in the presence of failures that affect the two modules under consideration, we can detect the presence of the failures. This detection is important for the system to take corrective action and directly affects the system availability as shown in Sec. 2.3. The fourth column of Table 5.1 shows the number of non-self-testable fault pairs in duplex systems with identical and different implementations. It is clear from Table 5.1 that with different implementations it is possible to achieve high self-testing properties of

the designs under consideration. In fact, an interesting synthesis problem is to synthesize two implementations of a given logic function such that the number of self-testable fault pairs is maximum.

Table 5.1. Self-testing properties of diverse and non-diverse duplex systems

Circuit Name	Copies	# SSF pairs	% escapes
Z5xp1	T, T	302,500	0.73
	C, C	372,100	0.65
	T, C	335,500	0.02
clip	T, T	487,204	0.58
	T, C	463,472	0.02
inc	T, T	236,196	0.73
	C, C	256,036	0.84
	T, C	245,916	0.03
rd84	T, T	322,624	0.74
	C, C	158,404	1.1
	T, C	226,064	0.04

6. Diversity Advantages In Configurable Systems

For evaluating the advantages of diversity in configurable systems, we implemented duplex systems on a configurable computing test-bed [Wildforce 99]. The test-bed contains FPGAs that can be used for mapping designs into the test-bed. Figure 6.1 shows the test-bed.

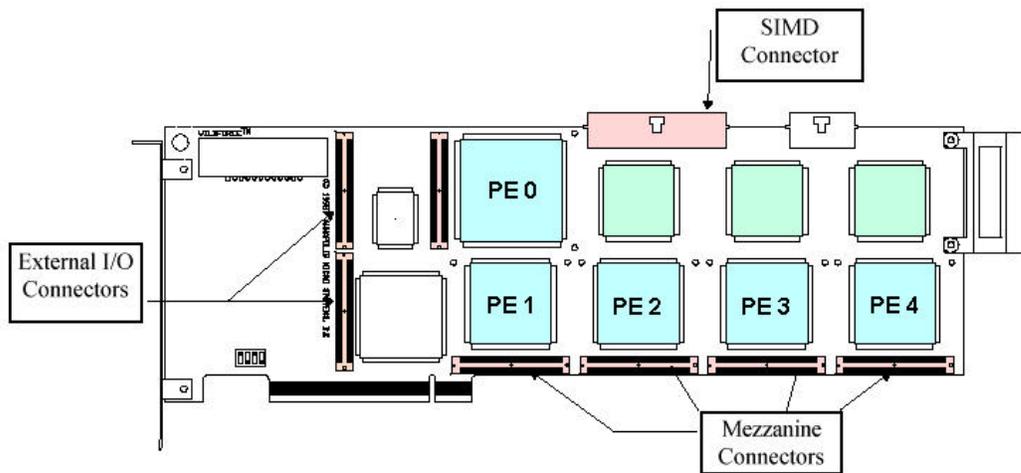


Figure 6.1. Reconfigurable computing test-bed

Duplex systems containing identical and different implementations of the same logic function were designed. The synthesis tool from Synplicity was used for synthesizing the different implementations. Xilinx placement and routing tools were used to map the designs on the test-bed. For each duplex system, we injected stuck-at

faults in the lookup tables of the implementations and for each fault pair, we calculated the error latency of that fault pair. We picked the worst-case fault pairs (just like Simulation 2) and plotted the cumulative distribution showing the percentage of worst-case fault pairs having the error latency less than or equal to a particular value.

In Fig. 6.2(a) we show the cumulative distribution of the worst-case error latencies for a duplex system with two identical implementations of the MCNC benchmark circuit *cps.pla* with 23 inputs. Note that, the X-axis is in the logarithmic scale. Figure 6.2(b) shows a similar cumulative distribution for a duplex system with different implementations of the same logic function (*cps.pla*). The faults were injected by modifying the contents in the FPGA lookup tables. We also calculated the mean error latency and it can be seen that the mean error latency is at least an order of magnitude greater for diverse duplex systems.

The significance of the curves in Fig. 6.2 can be explained with the help of the following example. Consider an application with a mission time of 10^6 cycles. For a system with identical implementations (Fig. 6.2(a)), the data-integrity of the system will be compromised before the mission-time is reached for around 85% of the cases in the presence of CMFs. In other words, for only around 15% of the CMFs, the system is expected to successfully complete the task before data corruption occurs. In contrast, if we use a duplex system with different implementations, then for around 65% of the cases, the system is expected to successfully finish the task before impacting data-integrity (Fig. 6.2(b)). This clearly demonstrates the advantage of diversity against CMFs and puts into perspective its relationship with the mission-time of the application, as explained in Sec. 2.2.

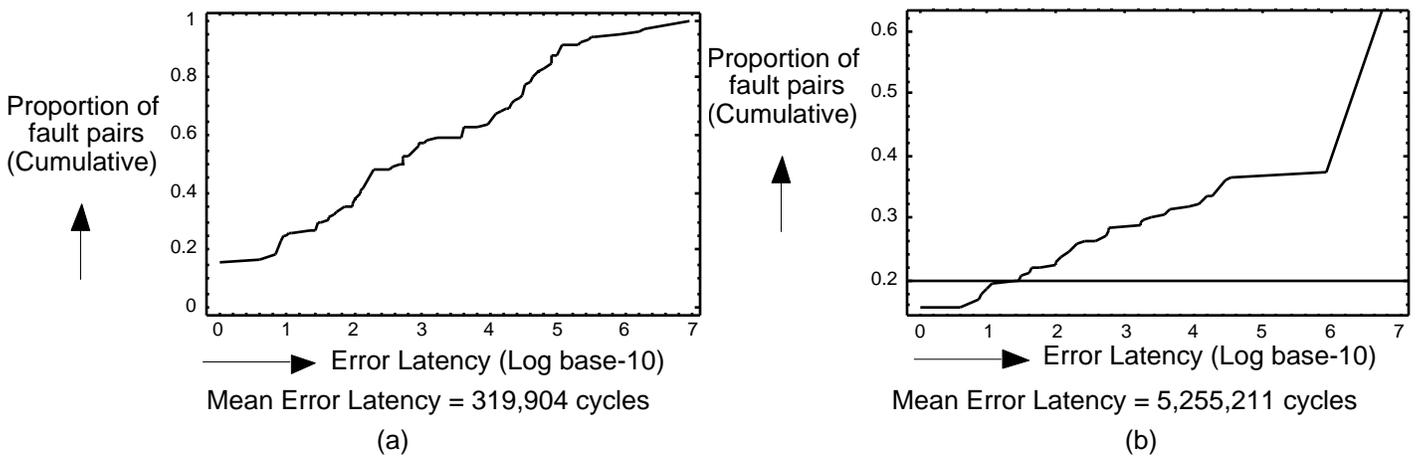


Figure 6.2. Results from experiments on a configurable computing test-bed
Some of the faults injected in the previous experiment are redundant faults.

Hence, the output data will never become corrupt in the presence of these faults. The plots in Fig. 6.3(a) and 6.3(b) show the statistics for the same experiments as above after excluding the redundant faults. It can be seen that the mean error latency of the diverse duplex system is around two orders of magnitude better than that of the duplex system with identical implementations. We have performed many such experiments and obtained similar data that demonstrate the advantages of using diverse duplex systems in configurable systems. While it is true that the actual nature of these curves depend on the sequence of input combinations applied, we performed multiple experiments with different input sequences and the curves show the same trend as those shown in Fig. 6.2 and Fig. 6.3.

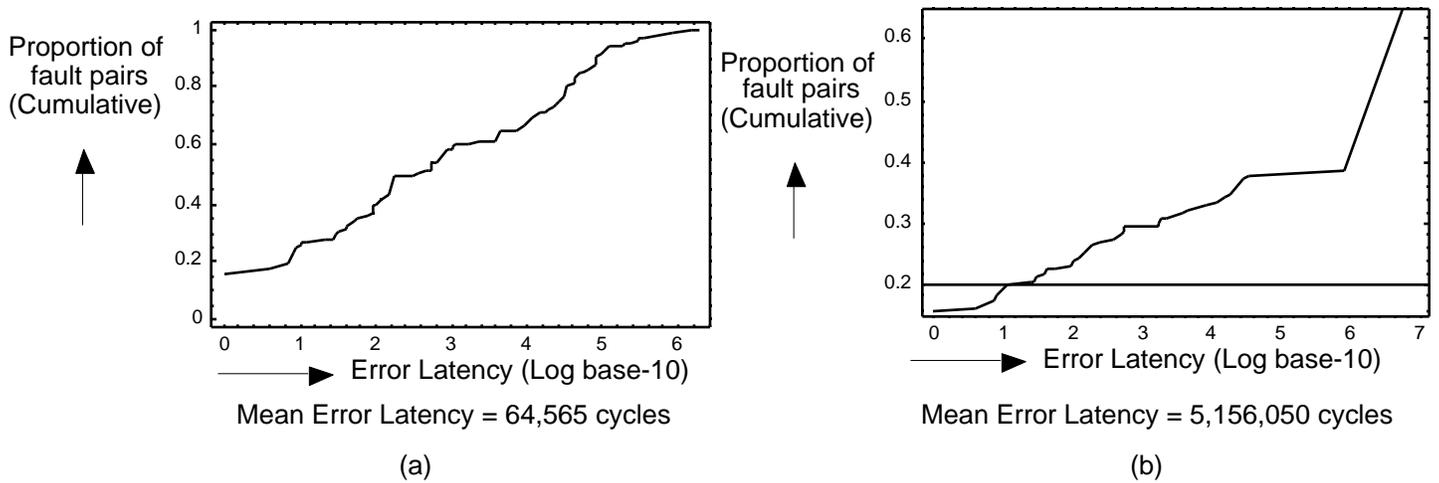


Figure 6.3. Results from experiments on a configurable computing test-bed (Excluding redundant faults)

7. Conclusions

In this paper, we addressed the problem of design diversity in redundant (software or hardware) systems in order to handle common-mode failures and failures in multiple modules. In order to protect fault-tolerant systems against common-mode failures, design diversity has been used commercially. In the past, design diversity was defined to be “independent” generation of “different” designs. This notion of diversity is qualitative and has limitations because it does not provide any quantitative basis to compare reliabilities of different diverse systems. Hence, the need for a metric to quantify diversity between different systems has been expressed in the past.

In this paper, for the first time, we have introduced a metric to quantify diversity among different designs under a particular fault-model, and explained how to calculate the overall system reliability in terms of this metric. In our example of the calculation of

PRELIMINARY VERSION

diversity for combinational logic circuits (Sec. 2.1), we have assumed that all the input combinations are equally likely. In the absence of any information about the relative frequency of the different input combinations, this is a reasonable assumption. However, for a particular application, if we have information about the relative frequencies (in the form of input traces, for example), then we can appropriately modify the above expression to incorporate this extra information (by changing the weights associated with different input combinations).

We have also produced simulation for duplex system. Our theoretical and simulation results indicate that, in the presence of independent multiple module failures in redundant systems, mere use of different implementations does not guarantee higher reliability compared to redundant systems with identical implementations. It is more important to evaluate the reliability of the systems using our metric. On the other hand, for common-mode failures and design faults, there is a significant gain with different implementations. However, the gain decreases with increasing mission time. Our analysis technique can be used to derive relationships between system reliability, diversity, mission time and system failure rate and compare reliabilities of multiple diverse systems. These relationships can help understand the cost and reliability tradeoffs while designing redundant systems with diversity.

For common-mode failures, diverse systems have no worse reliability compared to replicated systems. However, there is a further need to characterize common-mode failure mechanisms in the circuit level. With a good CMF fault model, (logical or layout-level) synthesis techniques can be used to incorporate sufficient diversity to protect systems against the modeled faults.

Our simulation results demonstrate that diversity plays an important role in enhancing the self-testing property of duplex systems. This can prove to be useful if we can apply specific patterns to the system during idle cycles.

8. Acknowledgments

This work was supported by Defense Advanced Research Project Agency (DARPA) under Contract No. DABT63-97-C-0024 (ROAR project). The authors would like to thank Dr. Santiago Fernandez Gomez of Stanford Center for Reliable Computing for his help with the experiments on the configurable computing test-bed. Thanks are due to Mr. Robert Wei-Je Huang and Mr. Philip Shirvani of Stanford Center For Reliable Computing.

9. References

[Avizienis 77] Avizienis, A. and L. Chen, "On the implementation of N-version programming for software fault-tolerance during program execution," *Proc. Intl. Computer Software and Appl. Conf.*, pp. 149-155, 1977.

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

[Briere 93] Briere, D. and P. Traverse, "Airbus A320/A330/A340 Electrical Flight Controls: A family of fault-tolerant systems," *Proc. FTCS*, pp. 616-623, 1993.

[Eckhardt 85] Eckhardt, D. E. and L. D. Lee, "A theoretical basis for the analysis of multi-version software subject to coincident failures," *IEEE Trans. Software Engg.*, Vol. SE-11, pp. 1511-1517, Dec. 1985.

[Gary 79] Gary, M. and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, 1979.

[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, January, 1994.

[Liu 97] Liu, J., et. al., "Heavy ion induced single event effects in semiconductor device," *Proc. Intl. Conference on Atomic Collisions in Solids*, 1997.

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

[Lyu 91] Lyu, M. R. and A. Avizienis, "Assuring design diversity in N-version software: a design paradigm for N-version programming," *Proc. DCCA*, pp. 197-218, 1991.

[Lyu 95] Lyu, M., *Handbook of Software Reliability Engineering*, Computer Society Press, 1995.

[McCluskey 71] McCluskey, E. J. and F. W. Clegg, "Fault Equivalence in combinational logic networks," *IEEE Trans. On Computers*, Vol. C-20, No. 11, pp. 1286-1293, Nov. 1971.

PRELIMINARY VERSION

[McCluskey 88] McCluskey, E. J., S. Makar, S. Mourad and K. D. Wagner, "Probability Models for Pseudo-random Test Sequences," *IEEE Trans. Computers*, Vol. 37, No. 2, pp. 160-174, Feb. 1988.

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

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

[Reed 97] Reed, R., et. al., "Heavy ion and proton-induced single event multiple upset," *IEEE Trans. on Nuclear Science*, Vol. 44, No. 6, pp. 2224-2229, July 1997.

[Riter 95] Riter, R., "Modeling and Testing a Critical Fault-Tolerant Multi-Process System," *Proc. FTCS*, pp. 516-521, 1995.

[Sakov 87] Sakov, J. and E. J. McCluskey, "Functional Test Pattern Generation for Random Logic," *CRC TR 87-1*, Center For Reliable Computing, Stanford Univ., 1987.

[Saxena 98] Saxena, N.R., and E.J. McCluskey, "Dependable Adaptive Computing Systems," *Proc. IEEE Systems, Man and Cybernetics Conf.*, San Diego, pp. 2172-2177, 1998.

[Shedletsky 76] Shedletsky, J.J., and E.J. McCluskey, "The Error Latency of a Fault in a Sequential Digital Circuit," *IEEE Trans. Computers*, C-25, No. 6, pp. 655-659, June 1976.

[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.

[Siewiorek 75] Siewiorek, D. P., "Reliability modeling of compensating module failures in majority voted redundancy," *IEEE Trans. Comp.*, vol. 24., no. 5, pp. 525-533, 1975.

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

PRELIMINARY VERSION

[Stroud 94] Stroud, C. E., "Reliability of Majority Voting Based VLSI Fault-Tolerant Circuits," *IEEE Trans. on VLSI*, vol. 2, no. 4, pp. 516-521, December, 1984.

[Tamir 84] Tamir, Y. and C. H. Sequin, "Reducing common mode failures in duplicate modules," *Proc. ICCD*, pp. 302-307, 1984.

[To 73] To, K., "Fault Folding for Irredundant and Redundant Combinational Circuits," *IEEE Trans. Comp.*, Vol. C-22, No. 11, pp. 1008-1015, Nov. 1973.

[Tohma 71] Tohma, Y. and S. Aoyagi, "Failure-tolerant sequential machines with past information," *IEEE Trans. Computers*, Vol. C-20, No. 4, pp. 392-396, April 1971.

[Tomek 93] Tomek, L. A., J. K. Muppala and K. S. Trivedi, "Modeling Correlation in Software Recovery Blocks," *IEEE Trans. Software Engg.*, Vol. 19, No. 11, pp. 1071-1086, Nov. 1993.

[Von Neumann 56] Von Neumann, J., "Probabilistic Logics and the synthesis of reliable organisms from unreliable components," *Automata Studies, Ann. of Math. Studies*, no. 34, pp. 43-98, 1956.

[Wildforce 99] *The Wildforce Board Manual*, Annapolis Microsystems, Inc., 1999.

APPENDIX

1. Reliability Calculation

In this section, we derive an expression for the probability that a duplex system produces correct outputs up to time t when two modules are affected by faults f_1 and f_2 , respectively. As explained in Sec. 3, there are two cases that must be considered. In the first case, the faults f_1 and f_2 appear simultaneously. Let p be the probability that a module is affected by a fault at any time instant. Also, we assume that only a single stuck fault can affect a particular module. With these assumptions, the probability that the fault pair appears at time instant i is given by the following expression:

$$(1 - p)^{2(i-1)} p^2$$

The above expression follows from the fact that the two modules are fault-free up to time $(i-1)$ and become faults at time instant i . Once the fault pair (f_1, f_2) arrives at time instant i , the probability that the system will produce correct outputs from time instant up

PRELIMINARY VERSION

to time t is obtained by multiplying $d_{1,2}$ $(t - i + 1)$ times to obtain the following expression:

$$(1-p)^{2(i-1)} p^2 d_{1,2}^{t-i+1}$$

In the above expression, i can vary from 1 to t . Thus, we have the following summation:

$$\sum_{i=1}^t (1-p)^{2(i-1)} p^2 d_{1,2}^{t-i+1}$$

The above summation evaluates to the following expression (by summation of Geometric Progression series):

$$s_1(f_1, f_2, t) = p^2 d_{1,2} \frac{[d_{1,2}^t - (1-p)^{2t}]}{[d_{1,2} - (1-p)^2]}$$

Next, we consider the case where the faults f_1 and f_2 do not appear simultaneously. First, we consider the case where fault f_1 appears earlier. The probability that faults f_1 and f_2 appear at times i and j , respectively ($j > i$) is given by:

$$(1-p)^{i-1} (1-p)^{j-1} p^2$$

In a duplex system, as long as two modules are working correctly, correct outputs are produced by the system. However, once fault f_2 affects Module 2, the probability that the system produces correct outputs starting from time j to time t can be obtained by multiplying the above expression by $d_{1,2}$ $(t - j + 1)$ times as shown below:

$$(1-p)^{i+j-2} p^2 d_{1,2}^{t-j+1}$$

Here, i can vary from 1 to $t-1$ and j can vary from $i+1$ to t . Thus, we have the following expression:

$$\sum_{i=1}^{t-1} \sum_{j=i+1}^t (1-p)^{(i+j-2)} p^2 d_{1,2}^{t-j+1}$$

In the above discussion, we assumed that f_1 appears before f_2 . In order to consider the other case, we can multiply the above expression by 2 to obtain the following expression:

$$s_2(f_1, f_2, t) = \frac{2}{(d_{1,2} - 1 + p)} (1-p) p^2 d_{1,2}^2 \frac{[d_{1,2}^{t-1} - (1-p)^{2t-2}]}{[d_{1,2} - (1-p)^2]} - \frac{2}{(d_{1,2} - 1 + p)} (1-p)^t p d_{1,2} [1 - (1-p)^{t-1}]$$

$s(f_1, f_2, t) = s_1(f_1, f_2, t) + s_2(f_1, f_2, t)$ is the probability that the system produces correct outputs when Module 1 is affected by fault f_1 and Module 2 is affected by fault f_2 .

PRELIMINARY VERSION

We can extend the above derivations for common-mode failures. For that purpose, we assume that p is the probability that a given pair of modules get affected. Let f be a common-mode failure that affects both the modules such that faults f_1 and f_2 appear in Module 1 and Module 2, respectively. The probability that the failure shows up at time instant i is given by the following expression:

$$(1 - p)^{i-1} p$$

Once f arrives at time instant i , the probability that the system will produce correct outputs from time instant up to time t is obtained by multiplying $d_{1,2}$ $(t - i + 1)$ times to obtain the following expression:

$$(1 - p)^{i-1} p d_{1,2}^{t-i+1}$$

Here, i can vary from 1 to t . Thus, we obtain the following expression:

$$z(f_1, f_2, t) = p d_{1,2} \frac{[d_{1,2}^t - (1 - p)^t]}{[d_{1,2} - (1 - p)]}$$

As mentioned in Sec. 2.2, since we are considering CMFs, we do not consider the case where f_1 and f_2 arrive at different times.